Annals of Mathematics and Artificial Intelligence (2020) 88:479–516
https://doi.org/10.1007/s10472-019-09654-6

Blending under deconstruction

The roles of logic, ontology, and cognition in computational concept invention

Roberto Confalonieri[1,2] · **Oliver Kutz**[1]

Published online: 25 July 2019

Abstract

The cognitive-linguistic theory of conceptual blending was introduced by Fauconnier and Turner in the late 90s to provide a descriptive model and foundational approach for the (almost uniquely) human ability to invent new concepts. Whilst blending is often described as 'fluid' and 'effortless' when ascribed to humans, it becomes a highly complex, multi-paradigm problem in Artificial Intelligence. This paper aims at presenting a coherent computational narrative, focusing on how one may derive a formal reconstruction of conceptual blending from a deconstruction of the human ability of concept invention into some of its core components. It thus focuses on presenting the key facets that a computational framework for concept invention should possess. A central theme in our narrative is the notion of refinement, understood as ways of specialising or generalising concepts, an idea that can be seen as providing conceptual uniformity to a number of theoretical constructs as well as implementation efforts underlying computational versions of conceptual blending. Particular elements underlying our reconstruction effort include ontologies and ontology-based reasoning, image schema theory, spatio-temporal reasoning, abstract specification, social choice theory, and axiom pinpointing. We overview and analyse adopted solutions and then focus on open perspectives that address two core problems in computational approaches to conceptual blending: searching for the shared semantic structure between concepts—the so-called generic space in conceptual blending—and concept evaluation, i.e., to determine the value of newly found blends.

Keywords Computational concept invention · Conceptual blending · Ontologies · Image schemas · Refinement operators

✉ Roberto Confalonieri
roberto.confalonieri@gmail.com

Oliver Kutz
oliver.kutz@unibz.it

[1] Faculty of Computer Science, Free University of Bozen-Bolzano,
Piazza Domenicani - Domenikanerplatz, 3, 39100 Bozen-Bolzano, Italy

[2] Present address: Telefónica Innovación Alpha, Plaça d'Ernest Lluch i Martin, 5,
E-08019 Barcelona, Spain

Mathematics Subject Classification (2010) 97R40 · 68T27

1 Introduction

How do you take a piece of raw fish, spaghetti, and some herbs, and turn it into a dish? How should you fill that empty canvas with colour? What makes a piano an instrument to play music on, and not a tool for cooking? What makes a question a part of a scientific inquiry? These, and many similar questions, have to do with the imaginative mind of humans, their ability to invent new concepts, and more broadly, the human potential for creativity.

Computational creativity is a multidisciplinary endeavour. Although the field started as part of research in Artificial Intelligence, it more recently also strongly overlaps and interacts with topics in cognitive linguistics and psychology, philosophy and the arts. Colton and Wiggins [16] give the following definition of the field: "The philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative." In practice, the goal of computational creativity is to model, simulate or replicate creativity using a computer, in order to achieve one of several ends: either to construct a program or computer capable of human-level creativity, or to better understand human creativity and to formulate an algorithmic perspective on creative behaviour in humans, or to design programs that can enhance human creativity without necessarily being creative themselves. As a field, computational creativity concerns itself with theoretical and practical issues in the study of creativity. Theoretical work on the nature and proper definition of creativity is performed in parallel with practical work on the implementation of systems that exhibit creativity, with one strand of work informing the other.

Computational creativity as an academic field in its own right has enjoyed growing popularity over the last decade since the First Joint Workshop on Computational Creativity in 2004.[1] Scientifically, papers reporting on work in computational creativity are by now part of the standard repertoire of major AI conferences and journals. With respect to the achieved results, much work was dedicated to the development of artificial systems that are capable of creating artistic artefacts in different domains: music, poetry, and paintings. These systems are the result of combining different AI techniques, both symbolic (e.g., automated theorem proving, ontologies), sub-symbolic (e.g., machine learning and deep learning), or in-between (e.g., case-based reasoning). Still, on the more cognitive system and problem-solving oriented side, progress has been slower and many questions concerning the cognitive nature and computational modelling of creativity, for instance in concept invention, idea generation, or inventive problem-solving, remain unanswered.

In the light of these recent achievements, in this paper, we study in what ways ontologies, and standard and non-standard reasoning tasks, support the realisation of a concept invention framework in a precise, formal and machine-computable way. In particular, we present

[1] The International Conference on Computational Creativity (ICCC) is now organised yearly, several FP7 FET-Open research projects were explicitly dedicated to questions in computational creativity (e.g., COINVENT [85], see also https://www.coinvent.uni-osnabrueck.de/en/home.html, dedicated to the topic of concept invention) and scientific symposia on creativity science were organised (e.g., the Symposium on Creativity Science at the ZiF Centre for Interdisciplinary Research, see https://www.uni-bielefeld.de/(en)/ZIf/AG/2016/09-19-Kuehnberger.html). Also, the FP7 3-year coordination action "Promoting the Scientific Exploration of Computational Creativity (PROSECCO)" was actively promoting computational creativity to the general public since 2013.

adopted solutions and open perspectives that address two open problems in computational approaches to conceptual blending: searching for the shared semantic structure between concepts—the so-called generic space in conceptual blending— and concept evaluation, namely how new blends can be evaluated.

An important approach contributing to an understanding of concept invention is the theory of *conceptual blending*, introduced by Fauconnier and Turner [27] which developed further the idea of *bisociation* introduced by the psychologist Koestler [52]. According to Fauconnier and Turner [27], conceptual blending is a cognitive process that serves a variety of cognitive purposes, including creativity. In this way of thinking, human creativity can be modelled as a blending process that takes different mental spaces as input and blends them into a new mental space called a blend. This is a form of combinational creativity, one of the three forms of creativity identified by Boden [13] (the other two being 'explanatory' and 'transformational'). A blend is constructed by taking the existing commonalities among the input mental spaces into account—known as the *generic space*—and by projecting the structure of the input spaces in a selective way (see Fig. 1). In general, the outcome can have an emergent structure arising from a non-trivial combination of the projected parts. Different projections lead to a large number of different blends and optimality principles are used to steer possible combinations that generate the blends. Conceptual blending has been successfully applied for describing existing blends of ideas and concepts in a varied number of fields, such as interface design, narrative style, poetry generation, mathematics, visual patterns, music theory, etc. [15, 27, 73, 77, 94, 95].

To illustrate the main idea behind conceptual blending, let us consider the following scenario:

> Maria is a chef in an Italian restaurant. One day her boss tells her that she needs to innovate the menu with something more *exotic*. She decides to *invent* a new cuisine style by *blending* the Italian and Peruvian cuisines. To this end, she explores the *combinations* of *seafood-spaghetti* with *ceviche*. After many attempts, she comes up with blended meals such as *spaghetti-with-ceviche*, *ceviche-with-spaghetti*, but also *sweet potatoes* with *spicy seafood*.[2]

The above scenario illustrates that, typically, the blended space inherits some features from either space and combines them to something novel. The blending of the input spaces involves a *generic space*, which contains the shared semantic structure between both input spaces. In the case of *seafood spaghetti* and *ceviche*, for instance, their generic space could be a *meal containing seafood*. The structure in the generic space is preserved in the blended space and specifics from the input spaces are projected into the blend in a meaningful way. This is not trivial because the naive combination of input spaces can lead to inconsistencies (and, more generally, to nonsense). For example, in the above blends, the *seafood* cannot be *raw* and *cooked* at the same time. Hence, before combining the input spaces into the *spaghetti-with-ceviche* concept, a *generalisation* is necessary, for instance, by *generalising* the *seafood* assignment of *cooked seafood* in the *seafood-spaghetti* concept.

Blending as a holistic human ability is effortless. We are able to create new blends spontaneously and have no difficulty to understand new conceptual blends when we encounter them. In contrast, the transposition of this cognitive theory into an automated system provides significant challenges and it requires a modular computation of different

[2] *Ceviche* is a dish made from fresh raw fish cured in lime juice with chili peppers and accompanied usually by sweet potatoes. The *spaghetti-ceviche* blend can be spaghetti with fresh raw fish, while the *ceviche-with-spaghetti* blend can be ceviche accompanied by spaghetti instead of sweet potatoes.

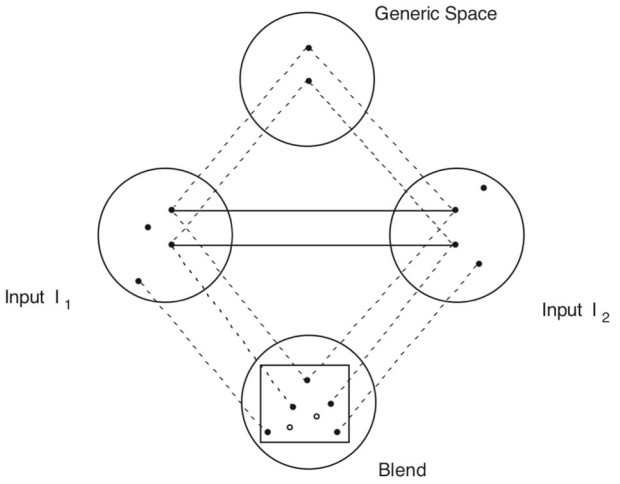

Fig. 1 The blending process as described by Fauconnier and Turner [26]

concept-invention tasks. These include the representation of the input spaces, the identification of a relevant generic space, the computation of the blend, and the evaluation of the usefulness of the blend. In the following, we will refer to these techniques as computational techniques for concept invention, and to any executable orchestration of them as a concept invention workflow. This paper focuses on the concept of refinement as a key approach to designing one complete workflow for conceptual blending. Here, refinements come in the form of syntactic operators that can both specialise and generalise the formal description of a concept, and they play a central role in almost all parts of the blending workflow.

The paper is structured as follows: Section 2 discusses in detail the computational approaches to blending followed in this paper, namely (i) Goguen's abstract model for blending, (ii) its computational interpretation in a workflow, (iii) the Amalgam-based interpretation of the Goguen model, and finally (iv) a summary of a number of techniques needed to make such a workflow work. Next, in Section 3, we first (v) briefly introduce the Description Logics $\mathcal{EL}^{++}$ and $\mathcal{ALC}$, followed by (vi) a detailed introduction and study of refinement operators (generalisation/specialisation) for these logics. Section 4 then addresses the crucial problem of finding the generic space for two different input concepts from two different perspectives, first by (vii) using the primarily syntactic generalisation operation within the amalgam-based workflow, and then (viii) refining the approach to work in orchestration with the cognitive theory of image schemas. Finally, Section 5 discusses the evaluation problem for blending from the specific point of view to employ axiom weakening for the purpose of 'repairing' a concept. This is studied (ix) in isolation as well as (x) in a variant build on social choice theory. The paper closes with a discussion and future perspectives.

2 Computational framework for concept invention

For Fauconnier and Turner, conceptual blending is a cognitive ability, and not a symbolic process. Nevertheless, it inspired researchers to search for computational representations of

conceptual blending. For a detailed overview of these frameworks we refer to [14, 21, 24]. In this paper, we focus on a blending framework built around knowledge representation techniques and corresponding enabling techniques.

2.1 Goguen's abstract model for conceptual blending

Goguen defines the approach of *algebraic semiotics* in which certain structural aspects of semiotic systems are logically formalised in terms of algebraic theories, sign systems, and their mappings [33]. Its main insight was that semiotic systems and conceptual spaces may be represented as logical theories, consisting of constants, relations, sorts, and axioms. If two such theories are related via *morphisms* to a third space (the generic space), then the blending of the input spaces is comparable to a *colimit computation*. This construction is comparable to a disjoint union modulo the identification of certain parts. In particular, the blending of two concepts is often a *pushout* [33].

In [35], algebraic semiotics has been applied to user interface design and conceptual blending. The approach of algebraic semiotics does not claim to provide a comprehensive formal theory of blending. Indeed, Goguen and Harrell admit that many aspects of blending cannot be captured formally. However, the structural aspects can be formalised and provide insights into the space of possible blends. The formalisation of these blends has been formulated using the algebraic specification language OBJ3 [37]. Since OBJ3 has been invented as a language for algebraic specification and programming, it is not best-suited for knowledge representation due to the lack of predicates, full Boolean connectives and existential quantifiers. Furthermore, conceptual blending is a process that is applicable to a wide range of domains and even across domains. However, the languages that are used to represent knowledge vary often significantly across domains; e.g., mathematical knowledge is usually represented using first- or higher-order formulas, abstract or taxonomic bio-medical knowledge is often represented in the Web Ontology Language (OWL) or in OBO [66], and music is often represented as notes on staves. Hence, one challenge is to develop a knowledge representation framework that enables the representation of the blending processes and that supports knowledge sources that are provided in a diverse set of knowledge representation languages.

2.2 Towards a computational framework for concept invention

Techniques for concept invention build on (and further develop) ontologies, used as a central tool in formalising a computational framework for conceptual blending [56–58]. Ontologies provide a shared and common understanding of a domain, and thus facilitate knowledge sharing and reuse. By making the conceptualisation of a domain explicit, ontologies allow one to overcome the problem of implicit and hidden knowledge. More recently, ontologies have also figured prominently in AI systems for computational creativity, in particular for implementations of conceptual blending [20, 57].

An ontological blending model, which follows the research direction of algebraic semiotics established by Goguen [33], has been proposed in [56–58]. In this model, Goguen's semiotic systems are replaced with modern ontology languages such as OWL (the Web Ontology Language) and Common Logic. In particular, conceptual spaces are represented as OWL ontologies, interlinked by the Distributed Ontology Language (DOL) [68]. Conceptual blends are modeled as blending diagrams that record the relationships between the generic space, the input conceptual spaces, and the blend. An extension of this ontological blending model, which also extends Goguen and Harrell [35]'s

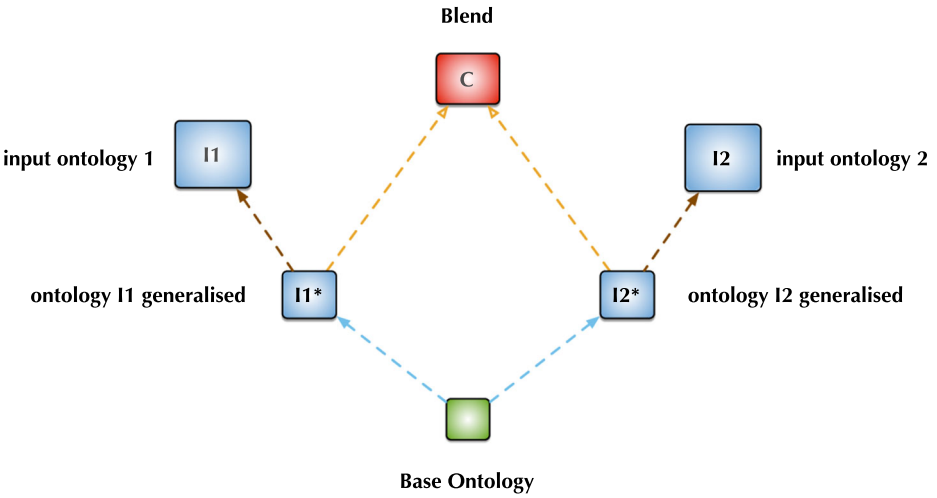

Fig. 2 The core model of computational conceptual blending

model, has been proposed in [85] as a core model of computational conceptual blending (see Fig. 2).[3]

To remove any doubt about our terminology, let us briefly discuss the way we, throughout this paper, use the terminology of 'theory', 'concept', and 'ontology': the formal objects that we associate with input (conceptual) spaces, base ontologies (generic spaces), and blends (or blended concepts, or blended theories, or sometimes 'blendoid' in the Goguen tradition), are logical theories specified according to some particular logical language L. In the literature, ontologies are typically described as documents or artefacts that are logical theories together with some additional annotations, most notably specific vocabularies and ontological commitments [32, 41, 69]. Since we here focus on the elements needed for a computational workflow for blending, these additional aspects will be, for now, of little significance (they do, or course, play a greater role in the evaluation problem for blends).[4] Moreover, when we say that we 'blend a concept', say the concept of *house*, what this means is that we operate with some ontology O_{house} in which the concept name *house* has been axiomatically defined, using a variety of other non-logical terms and the logical means available in L.

The core model differs from the ontological and Goguen and Harrell [35]'s models by introducing an extra step: the input ontologies I_1 and I_2 can also be generalised to two

[3]Notice that the blending diagram in Fig. 2 is an upside-down version of conceptual blending as visualised in Fig. 1, where the generic space is placed at the top identifying commonalities. The upside-down version follows Goguen and Harrell [35] with the generic space, or base ontology, at the bottom. This kind of diagram is, on the one hand, an upside-down version of the first illustration, following a tradition in mathematical diagrams where simpler concepts are often placed at the bottom. On the other hand, the term generic space is replaced with base ontology.

[4]A discussion of the impact of this distinction between theory and ontology on the practice of ontology engineering can be found in [69]. Note that, even in an expressive language L and given a rich axiomatisation O of a concept C, ontology O is thought to only approximately describe the intended models of C [41].

ontologies I_1^* and I_2^* (see Fig. 2).[5] There are several reasons why such a step might be necessary. First, when blending a concept from a given ontology, typically large parts of the ontology are in fact off-topic. Logically speaking, when extracting a module for the concept in question, large parts of the ontology turn out to be logically irrelevant. Second, when running the blend it may become obvious that the blend theory preserved too many properties from the input spaces — it may even become inconsistent. In this case, generalising the input spaces will lead to a better result.

The base ontology identifies some structure that is shared across I_1 and I_2, or, to put it differently, the generic space contains some theory, which can be found in both input spaces, but which abstracts from the peculiarities of the input spaces and generalises the theory in some domain-independent way.

The base ontology is a more general theory generated from the input ontologies, further restricted by the ways in which the input ontologies can be generalised into I_1^* and I_2^*. From a logical point of view, there exist two theory morphisms (interpretations of theories) which embed the base ontology into I_1^* and I_2^*. These interpretations are a key element to make the automatic blending process work.

The ontologies I_1^* and I_2^*, together with the base ontology and the two theory morphisms (interpretations) that connect the generic space to I_1^* and I_2^*, determine the blend. Informally, what happens is that the blend is a disjoint union of I_1^* and I_2^*, where the shared structure from the base is identified. Technically, this is a colimit computation, a construction that abstracts the operation of disjoint unions modulo the identification of certain parts specified by the base and the theory morphisms, as discussed in detail in [34, 55, 56]. Therefore, while the visualisation of the core model for blending as shown in Fig. 2 does not show any arrows pointing from the Base Ontology to the blend, the impact of the Base Ontology on possible blends is realised by the respective mappings from the base to the inputs, together with the semantics of the colimit operation, along the composition of the arrows. For a formal definition, see [14, 55].

The model provides a formal setting according to which blends are characterised, and it guarantees that for any input ontologies I_1 and I_2, there are many blends. In general there exists a plethora of options for generalising the input ontologies, for possible base ontologies and for the morphisms between the generic space ontology and the generalised input ontologies. However, most of these combinations lead to undesirable blends. The blend may be inconsistent, because it includes conflicting information from the input ontologies. Alternatively, the blend may contain too little information to be considered a coherent concept or just combine the wrong ideas to be of any use in a particular context.

Therefore, the major challenges for the implementation of a framework for conceptual blending are:

– Generic space: one of the key problems in conceptual blending is how to find the generic space between several input spaces. Usually, the generic space needed for the blend creation is provided manually. This relates to the problem of finding generalisations of the input spaces. Note that, following the cognitive theory of conceptual blending, the generic space comprises a common abstraction of the input concepts guiding the specifics of the blending. This abstraction need not be 'maximal' and there is

[5] Note that Goguen and Harrell achieved a similar effect by considering morphisms that do not preserve all axioms.

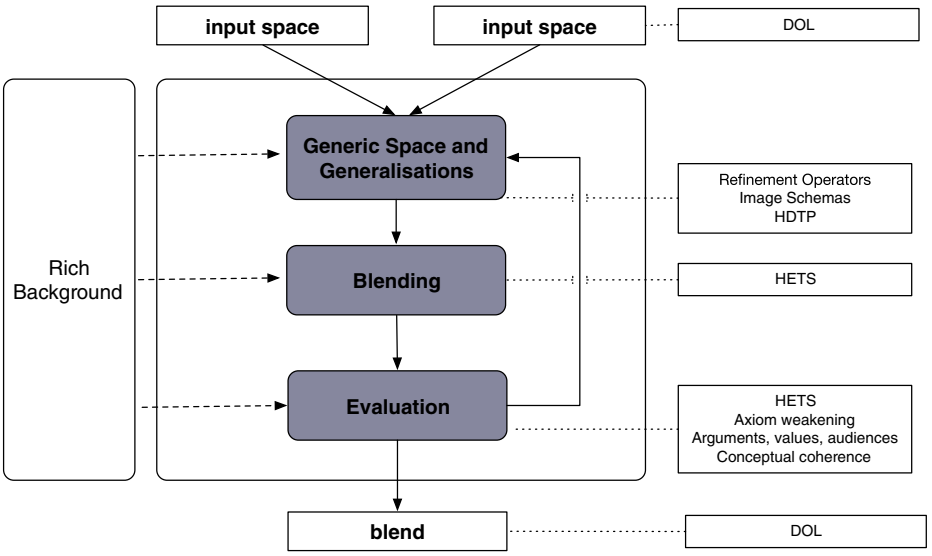

Fig. 3 Concept-invention workflow and enabling techniques

typically not a unique choice.[6] Studying the formal settings and enabling computational approaches to find generic spaces in an automatic or semi-automatic way is required in order to implement conceptual blending in a computational system exhibiting creativity.

– Blend evaluation: having established a generic space, and a set of generalisations, there typically remains a considerable number of combinations for the blend creation, and techniques for evaluating blended concepts need to be explored. Moreover, in the case of modeling blends as a logical theory, the blends generated can be logically inconsistent, but can still be interesting from a creative point of view. A mechanism to 'repair' them is needed.

At this point it should be clear that transposing conceptual blending into a computational setting for implementing a creative system is a complex task. One needs to try different combinations of generalised input theories, evaluation functions and criteria. Orchestrating all this requires a sophisticated concept invention workflow.

2.3 Concept invention workflow and techniques

A number of proposals for an implementation of concept invention workflows can already be found in the literature on (computational) conceptual blending [20, 24, 70, 79, 96, 97, 100]. Here, we describe the amalgam-based workflow implemented in [20, 22–24], which is a possible instantiation of the core model of conceptual blending presented in the previous section.

Figure 3 represents a dynamic view of how two input spaces are blended, in particular it shows how blend creation is an iterative process of blend generation and blend evaluation.

[6] This is why the generic space does not need to be a least general generalisation, as we shall see later in Section 3.2.

Blends are generated as *amalgams* [75], a notion developed in case-based reasoning as an approach to create a solution to a new case by combining multiple solutions of previous cases. According to this approach, input spaces are generalised until a generic space is found, and pairs of generalised versions of the input spaces are 'combined' to create blends.

Different strategies can be used to generate the generalised versions of the input spaces, namely, approaches based on anti-unification of theories such HDTP [89], refinement operators [20], or more cognitive-oriented approaches such as image schemas. Refinement operators allow one for the creation of weaker versions of certain concepts and/or axioms [20]. Image schemas, on the other hand, are representations of cognitive patterns that people learn in early childhood to structure their interpretations of perceptions, and which proved to be an important element in theories of embodied cognition and related computational approaches [46, 47].

The blend creation is achieved, as described in the previous sections, as a combination of a generalisation of the input spaces, and a projection of some of their specifics into a blended space. The way in which this combination is computed depends on the setting. In the case when formalising those elements in DOL, this is supported by means of the HETS tool [67].

Once a blend has been generated it needs to be evaluated. Blend evaluation is an important part of the blending itself, since not all the blends can be 'good' blends. What 'good' meas can depend on different aspects, being these cognitive optimality principles for blend creation or logically properties that a blend needs to satisfy when transposing conceptual blending in a (onto)logical setting.

To this end, it is assumed the existence of a rich background that contains background knowledge over the blending domain (or cross-domain), over which desirable properties, constraints and consequence requirements can be formulated. Constraints and consequence requirements are logical meta-statements that are supposed to be satisfied by the blended concepts. The former are statements that restrict the possible blends by requiring e.g., certain propositions *not* to hold, whilst the latter are statements that require the blend to entail certain properties (a detailed discussion can be found in [70]).[7]

A blend can be evaluated against logical consistency considering background knowledge. If the consistency check fails, the blend is repaired by analysing its internal structure and by looking for those axioms that are involved in the creation of the inconsistency. One or more axioms can be removed or, as we shall see, can be replaced by logically weaker versions, with the goal of preventing the inconsistency [93]. If the consistency test is successful, then it is evaluated whether the invented concept meets the consequence requirements for the blending process. Other options to evaluate blends are based on computational argumentation and conceptual coherence [17, 18, 86].

As it can be appreciated in Fig. 3, different seed techniques and computational frameworks can be adopted to implement the concept invention workflow.

– **The Distributed Ontology, Modeling and Specification Language:** The processes in Fig. 3 involve operations on documents that contain logical theories. This raises the question about the appropriate representation of these operations, and their semantics. To this end, the Distributed Ontology, Modeling and Specification Language (DOL) can be employed. DOL is an international ontology interoperability standard [74] that provides a unified metalanguage for handling heterogeneous knowledge [68]. It allows

[7] Such evaluation techniques for concepts are closely related to the technique of competency questions popular in ontology engineering [40, 83].

for a flexible modelling approach employing an open-ended number of formal logics. In particular, DOL provides constructs for (1) 'as-is' use of Ontologies, Models and Specifications (OMS) formulated in a specific ontology, modelling or specification language, (2) OMS formalised in heterogeneous logics, (3) modular OMS, (4) mappings between OMS, and (5) networks of OMS, including networks that specify blending diagrams. The DOL specification has been adopted as an international ontology interoperability standard. Among the languages supported by DOL are the Web Ontology Language OWL and the CASL logic [4], an extension of classical first-order logic.

- **Refinement Operators:** Refinement operators comprise a technique used to refine concepts and axioms into more general or more specific descriptions. Introduced in the Inductive Logic Programming literature to learn concepts from examples [61], refinement operators have been adapted in concept invention as a mechanism to compute blends as amalgams [20], refine image schemas [47], and implement ontology repair based on axiom weakening [81, 93].
- **Image schemas:** Image schemas are identified as fundamental pattern of cognition, that are perceived, conceptualised and manipulated spatially and temporally [65]. The main idea is that after an image schema has been formed, it can be generalised and the structure can be transferred through analogical reasoning to other domains with similar characteristics [65]. The relation of image schemas and conceptual blending—and, particularly, their role in computational concept invention—have been explored in [45, 47]. In this setting, image schemas are conceived as a set of theories ordered by logical entailment that can be used to guide the search for a generic space and as heuristics for the creation of new concepts.
- **Heuristic-Driven Theory Projection:** The Heuristic-Driven Theory Projection (HDTP) approach [89] comprises a reasoner that provides an implementation of analogical reasoning, strongly influenced by the earlier Structure Mapping Theory of Gentner [31]. According to this theory, analogical reasoning is characterised as the search for structural commonalities between a source and a target conceptualisation. These commonalities are associated with (mapped to) each other. HDTP is based on a restricted form of higher-order anti-unification and is able to compute the structural commonalities between two input domains represented in a variant of first-order logic.
- **HEterogeneous Tool Set:** The HEterogeneous Tool Set (HETS) system [67] is a parsing, static analysis and proof management tool incorporating various provers and different specification languages, thus providing a tool for heterogeneous specifications. It supports the colimit computation of CASL and OWL theories (the blend) and can check logical consistency using theorem provers such as *Eprover* [88] and *Darwin* [11].
- **Axiom weakening:** Axiom weakening is a novel technique for ontology repair according to which axioms that participate in causing an inconsistency are rewritten to a logically weaker version, instead of being removed [93]. The advantage is that whilst the technique can resolve inconsistencies, at the same time as much of the original knowledge resp. information content as possible is preserved.
- **Arguments, values and audiences:** Blend evaluation can be modeled using the notion of value-based argumentation framework [12]. Values represent qualities or properties that a blend should possess (e.g. following aesthetical considerations) and, given a blend, arguments promoting or demoting these values can be automatically generated and evaluated [18]. E.g., in the case of blending musical theories, values can be related to the tone and rhythm. Then, based on correspondingly generated arguments, decision making criteria can be specified to decide which blends to accept or to reject.

– **Conceptual coherence:** Blend evaluation can also be formulated as the problem of how newly invented concepts cohere w.r.t. the input spaces, generic space, and the rich background. The computational framework adopted here is the coherence theory by Thagard [92], in particular his notion of conceptual coherence. The basic intuition is that a blend is accepted or rejected depending on how much it contributes to maximising the constraints imposed by situational context and other relevant pieces of information modeled as axioms in an ontology [86].

Against the above mentioned background, it is clear that understanding, formalising, and implementing conceptual blending in computational creative systems is a major and challenging task. Different tools and frameworks are needed to realise the concept invention-related tasks in a computational way.

In the following, we will focus on two specific aspects of computational concept invention, namely, the search for a generic space, and the blend evaluation. We will do this from the perspective of refinement operators, and how these can be employed to define blends as amalgams, generalise input concepts and image schemas, and repair inconsistent blends. Without loss of generality, we frame these problems—and present their solutions—in the context of the description logics $\mathcal{EL}^{++}$ and $\mathcal{ALC}$, blending concepts specified in these logical formalisms.

3 The role of refinement operators in concept invention

3.1 Ontologies and the description logics $\mathcal{ALC}$ and $\mathcal{EL}^{++}$

From a formal point of view, an ontology is a set of formulas in an appropriate logical language with the purpose of describing a particular domain of interest, as previously discussed in Section 2.2. The precise logic used is in fact not crucial for our approach as most techniques presented here apply to a variety of logics; however, for the sake of clarity we use description logics (DLs) as well-known examples of ontology languages. We briefly introduce the basics for DL such as $\mathcal{ALC}$ and $\mathcal{EL}^{++}$; for full details see [6, 7]. The syntax of $\mathcal{ALC}$ is based on two disjoint sets N_C and N_R of *concept names* and *role names*, respectively. The set of $\mathcal{ALC}$ *concepts* is generated by the grammar

$$C ::= A \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall R.C \mid \exists R.C \ ,$$

where $A \in N_C$ and $R \in N_R$.

A *TBox* is a finite set of concept inclusions (GCIs) of the form $C \sqsubseteq D$ where C and D are concepts. It is used to store terminological knowledge regarding the relationships between concepts. An *ABox* is a finite set of formulas of the form $C(a)$ and $R(a, b)$, which express knowledge about objects in the knowledge domain.

The semantics of $\mathcal{ALC}$ is defined through *interpretations* $I = (\Delta^I, \cdot^I)$, where Δ^I is a non-empty *domain*, and $\cdot^I$ is a function mapping every individual name to an element of Δ^I, each concept name to a subset of the domain, and each role name to a binary relation on the domain. The interpretation I is a *model* of the TBox $\mathcal{T}$ if it satisfies all axioms in $\mathcal{T}$. Given two concepts C and D, we say that C is *subsumed* by D w.r.t. the TBox $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$) if $C^I \subseteq D^I$ for every model I of $\mathcal{T}$. We write $C \equiv_{\mathcal{T}} D$ when $C \sqsubseteq_{\mathcal{T}} D$ and $D \sqsubseteq_{\mathcal{T}} C$. C is *strictly subsumed by* D w.r.t. $\mathcal{T}$($C \sqsubset_{\mathcal{T}} D$) if $C \sqsubseteq_{\mathcal{T}} D$ and $C \not\equiv_{\mathcal{T}} D$.

$\mathcal{EL}^{++}$ is a DL that allows conjunctions, existential restrictions, and role inclusion axioms (RIs) of the form $r \sqsubseteq s$, where r and s are roles [7]. The bottom concept $\bot$, in combination with GCIs, allows one to express disjointness of concept descriptions, e.g., $C \sqcap D \sqsubseteq \bot$ tells that C and D are disjoint. Given two roles $r, s \in N_R$, we say that r is *subsumed* by s w.r.t. $\mathcal{T}$, denoted as $r \sqsubseteq_{\mathcal{T}} s$ if $r^I \subseteq s^I$ for every model I of $\mathcal{T}$. A role r is *strictly subsumed* by s w.r.t. $\mathcal{T}$ if $r \sqsubseteq_{\mathcal{T}} s$ and $r \not\equiv_{\mathcal{T}} s$.

$\mathcal{EL}^{++}$ is widely used in biomedical ontologies for describing large terminologies since subsumption (and thus classification) can be computed in polynomial time. In the following, $\mathcal{DL}$ denotes either $\mathcal{ALC}$ or $\mathcal{EL}^{++}$, and $\mathcal{L}(\mathcal{DL}, N_C, N_R)$ denotes the set of (complex) concepts that can be built over N_C and N_R in a given $\mathcal{DL}$.

3.2 Refinement operators

The refinement of formulas or concept descriptions has been studied both in the Description Logic and in the Inductive Logic Programming (ILP) literature, although from different perspectives. Whilst approaches in DL focus on the so-called 'non-standard reasoning tasks' of finding concept descriptions that are least general or most specific w.r.t. to a number of other concepts [9], approaches in ILP are concerned with learning DL descriptions from examples [63].

The general idea behind refinement is that given a set of formulas, specified according to a language $\mathcal{L}$ for which a *subsumption* relation ($\sqsubseteq$) is available, it is possible to modify them into expressions that are more general or more specific. Intuitively, the notions of generality and specificity can be introduced among any pair of formulas Ψ_1 and Ψ_2 by taking into account the subsumption relation. Therefore, also in $\mathcal{L}(\mathcal{DL}, N_C, N_R)$ with the subsumption relation $\sqsubseteq_{\mathcal{T}}$ for a given TBox $\mathcal{T}$.

The subsumption relation induces a partial order on the set of all formulas that can be formed using $\mathcal{L}(\mathcal{DL}, N_C, N_R)$, i.e., the pair $\langle \mathcal{L}(\mathcal{DL}, N_C, N_R), \sqsubseteq \rangle$ is a quasi-ordered set. In what follows, we say that a concept C_2 is more general than C_1 (resp. C_1 is more specific than C_2) if $C_1 \sqsubseteq C_2$.

Given the subsumption relation, for any two concepts C_1 and C_2, one can define the *least general generalisation* and *most general specialisation*. These have been relevant for defining semantic similarity measures between concepts [19, 76], and blends as an amalgam [20].

Definition 1 (**Least General Generalisation**) The least general generalisation of two concepts C_1 and C_2, denoted as $C_1 \blacktriangle C_2$, is defined as the most specific concept that subsumes both:

$$C_1 \blacktriangle C_2 = \{C \mid \ C_1 \sqsubseteq_{\mathcal{T}} C \wedge C_2 \sqsubseteq_{\mathcal{T}} C \text{ and } \nexists C' : C' \sqsubset_{\mathcal{T}} C \wedge C_1 \sqsubseteq_{\mathcal{T}} C' \wedge C_2 \sqsubseteq_{\mathcal{T}} C'\}$$

The least general generalisation is a description that encapsulates all the information that is common to both C_1 and C_2, and that is the most specific among those that subsume C_1 and C_2. The 'complementary' operation to the least general generalisation is the most general specialisation of two descriptions.

Definition 2 (**Most General Specialisation**) The most general specialisation of two concepts C_1 and C_2, denoted as $C_1 \blacktriangledown C_2$, is defined as the most general concept that is subsumed by both:

$$C_1 \blacktriangledown C_2 = \{C \mid C \sqsubseteq_{\mathcal{T}} C_1 \wedge C \sqsubseteq_{\mathcal{T}} C_2 \text{ and } \nexists C' : C \sqsubset C' \wedge C' \sqsubseteq_{\mathcal{T}} C_1 \wedge C' \sqsubseteq_{\mathcal{T}} C_2\}$$

If two descriptions have contradictory information, then they do not have a most general specialisation. The least general generalisation and the most general specification can be characterised by means of refinement operators.

Refinement operators are used to structure a search process for learning concepts from examples in Inductive Logic Programming [61]. In this setting, generalisation and specialisation refinement operators have been defined and used to build a refinement space (a lattice with $\top$ the most general element, and $\bot$ the more specific). Studying this notion in terms of generalisation and specialisation refinement operators has been done, for instance, for order sorted-feature terms clauses [76] and DL concepts [20, 63].

A refinement operator is a set-valued function that, given a certain concept, returns a set of concepts that are more general or more specific. Formally:

Definition 3 A generalisation refinement operator γ over $\langle \mathcal{L}(\mathcal{DL}, N_C, N_R), \sqsubseteq \rangle$ is a set-valued function such that $\forall C \in \mathcal{L}(\mathcal{DL}, N_C, N_R) : \gamma(C) \subseteq \{C' \in \mathcal{L}(\mathcal{DL}, N_C, N_R) \mid C \sqsubseteq_{\mathcal{T}} C'\}$.

Generalisation refinement operators take a concept C as input and return a set of descriptions that are more general than C by taking an ontology $\mathcal{T}$ into account. A specialisation operator, instead, returns a set of descriptions that are more specific.

Definition 4 A specialisation refinement operator ρ over $\langle \mathcal{L}(\mathcal{DL}, N_C, N_R), \sqsubseteq \rangle$ is a set-valued function such that $\forall C \in \mathcal{L}(\mathcal{DL}, N_C, N_R) : \rho(C) \subseteq \{C' \in \mathcal{L}(\mathcal{DL}, N_C, N_R) \mid C' \sqsubseteq_{\mathcal{T}} C\}$.

Refinement operators can be characterised according to some desirable properties, such as local finiteness, properness, and completeness [61, 63]. For instance, local finiteness requires that the set of refinements generated for any given element by the operator is finite; properness requires that none of the refinements generated by the operator is equivalent to the element being generalised or specialised, and, completeness requires that the refinement space contains all the formulas that can be constructed using the language $\mathcal{L}(\mathcal{DL}, N_C, N_R)$. Clearly, designing refinement operators that fulfil all these properties is not possible in general. One usually has to sacrifice completeness for finiteness to let the computation of the operator terminate, as we shall see in Section 4.1.

Refinement operators play an important role in computational approaches for conceptual blending. In particular:

Blends as amalgams and search for a generic space A key problem in computational conceptual blending is that the combination of two concepts may generate an unsatisfiable one due to contradiction, or the blended space may not satisfy certain properties and or constraints. The process according to which input concepts are blended is characterised using amalgams [75], a notion developed in case-based reasoning as an approach to combine solutions coming from multiple cases. The idea of amalgams is that, by refining input concepts, one can remove inconsistencies to find a novel and useful combination of the input concepts. For instance, a 'red French sedan' and a 'blue German minivan' can be blended to a 'red German sedan' by generalising the first concept to a 'red European sedan' and the second one to a 'coloured German car'. The least general generalisation of the input concepts—a 'coloured European car'—serves as an upper bound of the generalisation space to be explored, and plays the role of the so-called *generic space* in conceptual blending, which states the shared structure of both concepts. Nonetheless, the

least general generalisation might not always exist (see [20]), and for conceptual blending finding a common generalisation which is a least general generalisation is not required (and sometimes even not desirable). In fact, following the cognitive theory of conceptual blending, the generic space comprises a common abstraction of the input concepts guiding the blending, which, however, has to be in no way 'least general': an example are image schemas, which typically strip away far more information than would be logically necessary.

Refining image schemas In cognitive science, image schemas are identified as fundamental pattern of cognition, that are perceived, conceptualised and manipulated spatially and temporally [65]. The core idea is that after an image schema has been formed, it can be generalised and the structure can be transferred through analogical reasoning to other domains with similar characteristics [65]. The relation of image schemas and conceptual blending—and, particularly, their role in computational concept invention—has been explored in [45, 47]. In this setting, image schemas are conceived as a set of theories ordered by logical entailment that can be used to guide the search for a generic space and as heuristics for the creation of new concepts. Image schemas can offer high-level semantic constructs to identify the important parts of input concepts and ontologies, and they can be an effective way to reduce the search space of possible generic spaces. However, in some cases input concepts might not instantiate the same image schema, and image schemas need to be generalised into a more universal schema or need to be further specialised by adding new spatial primitives and axioms. Such relationships can be formally established semantically by interpretations of theories, and syntactically by building a refinement path. The mechanics of generalisation and specialisation for image schemas are discussed in detail in Section 4.2.2.

Blend evaluation and debugging In the context of using ontologies as formal backbones to represent blended concepts, a key problem is how to deal with the resulting formal inconsistencies, requiring an evaluation and debugging workflow. Handling inconsistency in concept invention is a complex issue, since an inconsistent concept can be still interesting if it can be 'repaired'. In this respect, a number of ontology debugging techniques have been developed, in particular in the Semantic Web community. They are typically based on computing (all or a subset of all) minimally inconsistent subsets for finding explanations, or on axiom pinpointing to remove a few axioms to expel the errors [8, 50, 51, 84]. While these methods are effective, and have been used in practice, they have the side effect of removing also many potentially wanted implicit consequences, and potentially interesting characteristics of the blend. Refinement operators were used to define a more fine-grained method for ontology repair based on axiom weakening [93], thus generalising earlier methods.

In the following sections, we will consider each of these roles in a more detailed way, presenting their application in concept invention.

4 Finding the semantic pathway for concept invention

There are different approaches in the AI and cognitive science literature that relate to the problem of finding the structural commonalities between two input domains, concepts or ontologies. Here, we describe approaches based on amalgams and refinement operators, and on image schemas.

4.1 Finding a generic space using generalisation operators

In this section we describe how blends can be characterised via amalgams and how a generic space between concepts can be found by means of a generalisation refinement operator in the DL $\mathcal{EL}^{++}$.

4.1.1 Blend as amalgams

The process of conceptual blending can be characterised via amalgams [75]. According to this approach, input concepts are generalised until a generic space is found, and pairs of generalised versions of the input concepts are 'combined' to create blends.

Formally, the notion of amalgams can be defined in any representation language $\mathcal{L}$ for which a subsumption relation $\sqsubseteq$ between formulas (or descriptions) of $\mathcal{L}$ can be defined, together with a least general generalisation operation, and a most general specialisation operation (see Definitions 1 and 2). Here, the least general generalisation is used to determine the generic space.

An *amalgam* of two descriptions is a new description that contains *parts from these two descriptions*. Formally, an amalgam is defined as follows.

Definition 5 (**Blend as an Amalgam**) Let $\mathcal{T}$ be an $\mathcal{EL}^{++}$ TBox. A description $C_b \in \mathcal{L}(\mathcal{EL}^{++}, N_C, N_R)$ is an *amalgam* of two descriptions C_1 and C_2 if there exist two generalisations $\overline{C_1}$ and $\overline{C_2}$ such that:

1. $C_1 \sqsubseteq_{\mathcal{T}} \overline{C_1} \sqsubseteq_{\mathcal{T}} C_1 \blacktriangle C_2$,
2. $C_2 \sqsubseteq_{\mathcal{T}} \overline{C_2} \sqsubseteq_{\mathcal{T}} C_1 \blacktriangle C_2$, and
3. $C_b = \overline{C_1} \blacktriangledown \overline{C_2}$.

The above definition relies on the least general generalisation $C_1 \blacktriangle C_2$ (Definition 1) that is a lower bound on the space of generalisations that can be explored for C_1 and C_2. At this point we should notice that finding a least general generalisation is usually complicated. Its computation (and existence) depends on the type of logic adopted and on the assumptions made over the TBox. Thus, in practice, finding a generic space between two concepts that is also a least general generalisation is not always possible. In conceptual blending, although one needs to be aware of properties shared by the concepts in order to blend them, it is not necessary to find a generic space that is also a least general generalisation. Namely, this choice would guarantee to preserve all common structure between the input concepts. Rather, a common subsumer w.r.t. the subconcepts that can be built using the axioms in a TBox, and sufficiently specific, will suffice. This can be achieved, as we shall see, by restricting the space of generalisations (Fig. 4).

$\mathsf{Clade} \sqsubseteq \top$	,	$\mathsf{LayEggs} \sqsubseteq \mathsf{Ability}$
$\mathsf{Avialae} \sqsubseteq \mathsf{Clade}$	,	$\mathsf{Walk} \sqsubseteq \mathsf{Ability}$
$\mathsf{Mammal} \sqsubseteq \mathsf{Clade}$	,	$\mathsf{Trot} \sqsubseteq \mathsf{Ability}$
$\mathsf{Ability} \sqsubseteq \top$	,	$\mathsf{BodyPart} \sqsubseteq \top$
$\mathsf{Fly} \sqsubseteq \mathsf{Ability}$	,	$\mathsf{Legs} \sqsubseteq \mathsf{BodyPart}$
$\mathsf{Mammal} \sqcap \exists \mathsf{hasAbility}.\mathsf{LayEggs} \sqsubseteq \bot$	,	$\mathsf{Torso} \sqsubseteq \mathsf{BodyPart}$
$\mathsf{Avialae} \sqcap \exists \mathsf{hasAbility}.\mathsf{Trot} \sqsubseteq \bot$	,	$\mathsf{Wings} \sqsubseteq \mathsf{BodyPart}$
$\mathsf{Domain}(\mathsf{hasAbility}) = \mathsf{Clade}$	,	$\mathsf{Range}(\mathsf{hasAbility}) = \mathsf{Ability}$
$\mathsf{Domain}(\mathsf{hasBodyPart}) = \mathsf{Clade}$	,	$\mathsf{Range}(\mathsf{hasBodyPart}) = \mathsf{BodyPart}$

Fig. 4 An ontology for the concept Horse and Bird

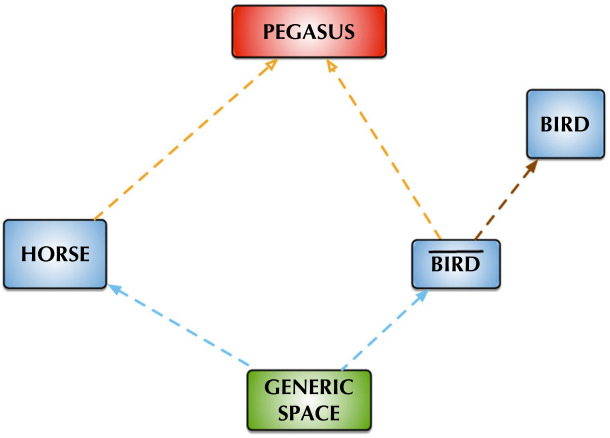

Fig. 5 A diagram of an (asymmetric) amalgam Pegasus from descriptions Horse and $\overline{\text{Bird}}$

Returning to Definition 5, the notions of amalgam and generic space are illustrated in Fig. 5 by means of a typical blend example: Pegasus, the winged divine stallion. From a conceptual blending point of view, Pegasus is a blend between a horse and a bird, preserving most of the horse's physical characteristics but adding bird-like features such as the wings and the ability to fly. A *horse* and a *bird* can be formalised by concepts describing different subtypes of *clade*, and involving some specific *body-parts* and *abilities*. For instance, a horse is a *mammal*, with a *torso* and *legs*, and with the *ability* to *walk* and to *trot*. This ontology is shown in Fig. 4. A stereotypical characterisation of a horse and a bird is:

$$\begin{aligned} \mathsf{Horse} \sqsubseteq\ & \mathsf{Mammal} \sqcap \exists \mathsf{hasBodyPart.Torso} \sqcap \exists \mathsf{hasBodyPart.Legs} \sqcap \\ & \exists \mathsf{hasAbility.Walk} \sqcap \exists \mathsf{hasAbility.Trot} \\ \mathsf{Bird} \sqsubseteq\ & \mathsf{Avialae} \sqcap \exists \mathsf{hasBodyPart.Torso} \sqcap \exists \mathsf{hasBodyPart.Legs} \sqcap \\ & \exists \mathsf{hasBodyPart.Wings} \sqcap \exists \mathsf{hasAbility.LayEggs} \sqcap \exists \mathsf{hasAbility.Fly} \end{aligned}$$

The combination of these concepts violates the common sense knowledge that mammals do not generally lay eggs and that avialae do not trot.[8] Therefore, these abilities need to be generalised before these concepts can be blended. The common descriptions between a horse and a bird—a clade with body-parts torso and legs and some abilities—defines a lower bound in the space of generalisations that can be explored in order to generalise the input concepts. These generalised concepts, in turn, can be used to find the blend of Pegasus. A generic space for the concepts Horse and Bird is:

$$\begin{aligned} & \mathsf{Clade} \sqcap \exists \mathsf{hasBodyPart.Torso} \sqcap \exists \mathsf{hasBodyPart.Legs} \sqcap \\ & \exists \mathsf{hasAbility.Ability} \sqcap \exists \mathsf{hasAbility.Ability} \end{aligned}$$

[8] This common sense knowledge can be modeled by disjointness axioms, such as $\mathsf{Mammals} \sqcap \exists \mathsf{hasAbility.LayEggs} \sqsubseteq \bot$ and $\mathsf{Avialae} \sqcap \exists \mathsf{hasAbility.Trot} \sqsubseteq \bot$. This axioms will 'invalidate' the blend theory when one checks for acceptability of concept blends based on their consistency. For the sake of this example, we do not consider the case of the *platypus*, an egg-laying mammal.

and it can be obtained as follows. In the Horse concept, Mammal is generalised to Clade, ∃hasAbility.Walk to ∃hasAbility.Ability, and ∃hasAbility.Trot to ∃hasAbility.Ability. In the Bird concept, Avialae is generalised to Clade, ∃hasAbility.LayEggs to ∃hasAbility.Ability, ∃hasAbility.Fly to ∃hasAbility.Ability, wheras the relation ∃hasBodyPart.Wings will be 'removed'. A generalised version of the bird concepts is:

$$\overline{\mathsf{Bird}} \sqsubseteq \quad \mathsf{Clade} \sqcap \exists \mathsf{hasBodyPart.Torso} \sqcap \exists \mathsf{hasBodyPart.Legs} \sqcap \exists \mathsf{hasBodyPart.Wings} \sqcap \exists \mathsf{hasAbility.Fly}$$

When we blend $\overline{\mathsf{Bird}}$ with Horse, we obtain a concept describing Pegasus. Please notice that in this case we can use a special case of amalgam (called asymmetric amalgam), in which $\overline{\mathsf{Horse}}$ and Horse coincide.

$$\mathsf{Mammal} \sqcap \exists \mathsf{hasBodyPart.Torso} \sqcap \exists \mathsf{hasBodyPart.Legs} \sqcap \exists \mathsf{hasBodyPart.Wings} \sqcap \exists \mathsf{hasAbility.Walk} \sqcap \exists \mathsf{hasAbility.Trot} \sqcap \exists \mathsf{hasAbility.Fly}$$

The generalisations above can be formally obtained by the application of a generalisation operator that is defined on the structure of concept definitions in an inductive way. Therefore, the definition of these operators depends on the description logic adopted. For the case of the DL $\mathcal{EL}^{++}$ [7], for instance, the generalisation operator was defined in [20].

4.1.2 Generalisation operator

A generalisation refinement operator for $\mathcal{EL}^{++}$ was defined in [20] by specifying three things: i) the (finite) set of subconcepts that can be formed from a TBox $\mathcal{T}$; ii) the upward cover set of a concept C and a role r, and iii) a set of transformation rules defined over the structure of concept descriptions.

The finite set of subconcepts that can be built from the axioms of a TBox $\mathcal{T}$ is obtained by structural induction over the concept descriptions. The set is denoted by $\mathsf{sub}(\mathcal{T})$.

Definition 6 Let $\mathcal{T}$ be an $\mathcal{EL}^{++}$ TBox. The set of *subconcepts* of $\mathcal{T}$ is given as

$$\mathsf{sub}(\mathcal{T}) = \{\top, \bot\} \cup \bigcup_{C \sqsubseteq D \in \mathcal{T}} \mathsf{sub}(C) \cup \mathsf{sub}(D) \tag{1}$$

where sub is inductively defined over the structure of concept descriptions as follows:

$$\begin{aligned} \mathsf{sub}(A) &= \{A\} \\ \mathsf{sub}(\bot) &= \{\bot\} \\ \mathsf{sub}(\top) &= \{\top\} \\ \mathsf{sub}(C \sqcap D) &= \{C \sqcap D\} \cup \mathsf{sub}(C) \cup \mathsf{sub}(D) \\ \mathsf{sub}(\exists r.C) &= \{\exists r.C\} \cup \mathsf{sub}(C) \end{aligned}$$

Based on $\mathsf{sub}(\mathcal{T})$, we define the upward cover set of atomic concepts and roles. $\mathsf{sub}(\mathcal{T})$ guarantees the following upward cover set to be finite.[9] The upward cover sets of a concept C and a role r ($\mathsf{UpCov}_{\mathcal{T}}$) contain the most specific subconcepts and roles found in $\mathsf{sub}(\mathcal{T})$ that are more general than (subsume) C and r respectively.

[9] We assume that $\mathcal{T}$ is finite.

Definition 7 (**Upcover sets**) Let $\mathcal{T}$ be an $\mathcal{EL}^{++}$ TBox, C a concept, and r a role. The *upward cover* sets of C and r w.r.t. $\mathcal{T}$ are:

$$\begin{aligned}\mathsf{UpCov}_{\mathcal{T}}(C) &:= \{D \in \mathsf{sub}(\mathcal{T}) \mid C \sqsubseteq_{\mathcal{T}} D \text{ and} \\ &\quad \nexists.\ D' \in \mathsf{sub}(\mathcal{T}) \text{ with } C \sqsubset_{\mathcal{T}} D' \sqsubset_{\mathcal{T}} D\}, \\ \mathsf{UpCov}_{\mathcal{T}}(r) &:= \{s \in N_r \mid r \sqsubseteq_{\mathcal{T}} s \text{ and} \\ &\quad \nexists.\ s' \in N_r \text{ with } r \sqsubset_{\mathcal{T}} s' \sqsubset_{\mathcal{T}} s\}.\end{aligned}$$

Based on the previous definitions a generalisation operator for $\mathcal{EL}^{++}$ concepts can be defined as follows.

Definition 8 (**Generalisation operator**) Let $\mathcal{T}$ be an $\mathcal{EL}^{++}$ TBox. A *generalisation refinement operator* γ is defined inductively over the structure of concept descriptions as follows:

$$\begin{aligned}\gamma(A) &= \mathsf{UpCov}_{\mathcal{T}}(A) \\ \gamma(\top) &= \mathsf{UpCov}_{\mathcal{T}}(\top) \\ \gamma(\bot) &= \mathsf{UpCov}_{\mathcal{T}}(\bot) \\ \gamma(C \sqcap D) &= \{C' \sqcap D \mid C' \in \gamma(C)\} \cup \{C \sqcap D' \mid D' \in \gamma(D)\} \cup \mathsf{UpCov}_{\mathcal{T}}(C \sqcap D) \\ \gamma(\exists r.C) &= \{\exists r.C' \mid C' \in \gamma(C)\} \cup \{\exists r'.C \mid r' \in \mathsf{UpCov}_{\mathcal{T}}(r)\} \cup \mathsf{UpCov}_{\mathcal{T}}(\exists r.C)\end{aligned}$$

Notice that by the repetitive application of γ, an $\mathcal{EL}^{++}$ concept C can be eventually generalised to $\top$. The idea behind computing the generic space between two concepts is to generalise both of them until a certain common concept is found.

To introduce the notion of generic space we define the set of all concepts that can be reached from C by applying γ a finite number of times.

Definition 9 For every concept C, we note $\gamma^i(C)$ the i-th iteration of its generalisation. It is inductively defined as follows:

- $\gamma^0(C) = \{C\}$;
- $\gamma^{j+1}(C) = \gamma^j(C) \cup \bigcup_{C' \in \gamma^j(C)} \gamma(C')$, $j \geq 0$.

Definition 10 The set of all concepts that can be reached from C by means of γ in a finite number of steps is:

$$\gamma^*(C) = \bigcup_{i \geq 0} \gamma^i(C) \ .$$

Every concept C can be generalised at most until the $\top$ concept. Thus, in the above formula, the finite number of steps corresponds to the number of generalisations needed to reach $\top$, and this number depends on the ontology and on the structures of the axioms contained in the ontology.

Having a way to generalise concepts in a finite way, we can now define the notion of generic space of the concepts. The generic space between two concepts is the most specific concept among the set of generalisations that can be obtained. It can be characterised as follows.

Definition 11 Let C and D be concept descriptions, the generic space of C and D is a concept description G such that the following conditions hold:

1. $G \in \gamma^*(C) \cap \gamma^*(D)$, and
2. $\nexists G'.G' \in \gamma^*(C) \cap \gamma^*(D)$ such that $G' \sqsubset_{\mathcal{T}} G$

Finding the generic space between concepts was implemented in [20] for the $\mathcal{EL}^{++}$ description logic using Answer Set Programming (ASP) [30], a well-known declarative programming paradigm to solve non-monotonic search problems. A domain-independent ASP program generalises $\mathcal{EL}^{++}$ concepts in a step-wise transition process. To this end, each step of the generalisation refinement operator in Definition 8 was implemented as an action by means of *precondition*, *inertia*, and *effect* rules. Following the ASP incremental approach to planning problems [29], a domain-independent ASP program was specified to search for applicable generalisation steps. Then, this program is instantiated with domain knowledge, that is obtained by translating the $\mathcal{EL}^{++}$ TBox into ASP facts and predicates. $\mathcal{EL}^{++}$ concepts are generalised until their descriptions are equal. The stable models of the ASP program contain the generalisation steps to be applied in order to generalise the $\mathcal{EL}^{++}$ concepts until a generic space is reached. Further details about the implementation can be found in [20].

Although using refinement operators is an interesting approach to find the generic space between two (or more) input concepts, at this point we should note that computing the space of generalisations can be costly (in terms of computation), especially, in the case of more expressive description logics where the subsumption reasoning task is not polynomial as in the case of $\mathcal{EL}^{++}$. Furthermore, when implementing the operators, other considerations must be made regarding their finiteness. It can happen indeed that, although each generalisation step is finite (the so-called locally finite property), the set of all generalisations is infinite. For instance, this is the case for cyclic TBoxes where some axioms can lead to infinite generalisation paths. Preventing this amounts to putting a limit on the number of nested generalisations allowed to be made. This is typically done by counting the depth-level of generalisations [20]. It is unclear, however, how an optimal level for this upper bound can be determined in general. On the one hand, with a low value, interesting generalisations can be missed; conversely, a high value can lead to a generalisation blow-up making conceptual blending computationally much harder.

Furthermore, apart from finiteness, typically, for any two input concepts there exists a large number of potential generalisations. Therefore, the search space for potential generic spaces (and, consequently, for potential conceptual blends) is vast. As a result, an increase of the size of the ontology leads to an explosion of possibilities for generalisations. Although some heuristics can be defined, as discussed above in connection with 'optimality principles' for blending, there is no purely formal way to distinguish cognitively relevant from irrelevant information.

In purely formal, logical approaches, the idea of relevant implication can be used to determine which information is relevantly entailed by other information, for instance, by requiring the sharing of information via sharing of symbols between antecedent and consequent, or by modalising to strict implication, see e.g., [1, 64]. Such considerations give rise to interesting directions for a further development of refinement operators, namely to relevant refinements. However, even a formal theory of relevant refinement in such a setup does not solve the problem of cognitive relevance.

Although we here refrain from trying to give a formal account of the notion of 'cognitive relevance', some of its basic elements will include 'human comprehensibility', 'informativeness', or 'inferential usefulness', following in the footsteps of the Gricean maxims [38] or more generally Sperber and Wilson's relevance theory [99].

In the following, we want to outline such a more cognitively-driven approach, namely using the idea of image schemas as a guide to find generic spaces.

4.2 Finding a generic space using image schemas

We have discussed an approach to finding the generic space based on amalgams and generalisations. The core underlying idea was that to identify non-trivial shared information content between two concepts, a careful generalisation process can be used, which we based on the above introduced generalisation operators.

The approach in this section can be seen as a variation or refinement of these ideas, namely that by trying to steer this generalisation process not to identify arbitrary 'shared information content or structure', but towards identifying shared semantically and cognitively crucial aspects of the input concepts, more informative and surprising blends become possible.

4.2.1 Image schemas and concept invention

In cognitive science, image schemas are identified as fundamental pattern of cognition that are perceived, conceptualised and manipulated spatially and temporally [65]. Prominent examples of image schemas introduced in the literature include SOURCE_PATH_GOAL (discussed in more detail below) CONTAINMENT, and SUPPORT. The core idea is that after an image schema has been formed, it can be generalised and the structure can be transferred through analogical reasoning to other domains with similar characteristics [65]. From a formal perspective, previous research on image schemas (e.g., [53, 91, 98]) has provided a valuable portfolio of approaches that can be build on further. The relation of image schemas and conceptual blending—and, particularly, their role in computational concept invention—have been explored in [45, 47]. In this setting, image schemas are conceived as a set of theories ordered by logical entailment that can be used to guide the search for a generic space and as heuristics for the creation of new concepts.

Although there is generally an increasing interest in the area of cognitively motivated Artificial Intelligence, where image schemas are suggested to be a core piece in the puzzle to model human-level conceptualisation and reasoning, so far rather few formal logical approaches can be found in the literature, in particular regarding attention to the dynamic aspects of image schemas. A fundamental problem here is that the typical mainstream approaches in contemporary KR do not map well to various scenarios found in image schema modelling. Therefore, a fine-tuned logic for image schemas was developed in [48], called ISL, essentially being a spatio-temporal logic for 'directed movement of objects', with the aim to model formally image schematic events such as 'Blockage', 'Caused Movement' and 'Bouncing'. It combines the Region Connection Calculus (RCC-8), Qualitative Trajectory Calculus (QTC), Cardinal Directions (CD) and Linear Temporal Logic over the reals (RTL). A further extension adding a notion of agency based on STIT logic was proposed in [60].

Hedblom et al. [45] provided an extensive motivation for treating image schemas not in isolation but as families of theories, organised in a lattice-like fashion. We here illustrate this by looking at some examples for concepts which involve members of the PATH-following family.

The most straightforward examples of concepts that involve PATH-following are concepts that are about the spatial relationship of movement between different points. Prepositions such as *from, to, across* and *through* all indicate a kind of PATH-following.[10]

[10] Some prepositions include other image schemas at the same time. E.g., 'through' involves apart from PATH also some notion of CONTAINMENT.

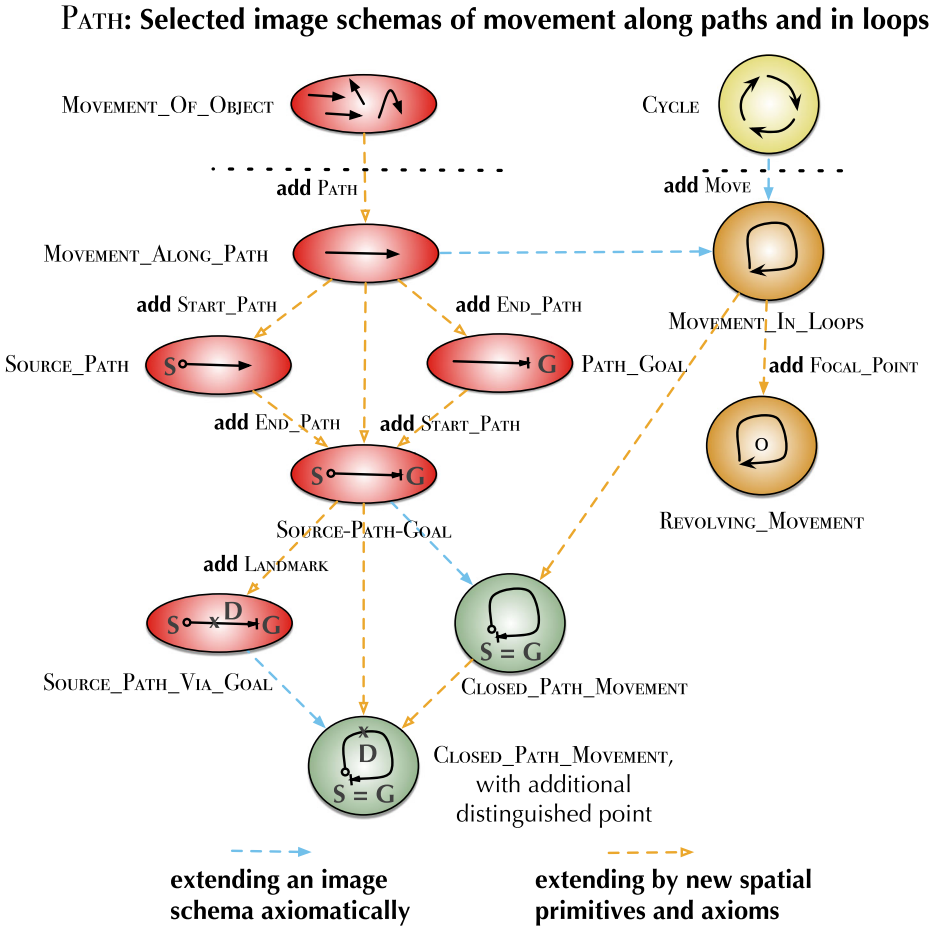

Fig. 6 Selected image schemas of path and cyclic movement as a graph. The coloured arrows described as "extending an image schema axiomatically" respectively "extending by new spatial primitives and axioms" illustrate by which means the PATH family is formally extended

This also includes key verbs that describe movement, e.g., *coming* and *going*. Another example, here for the image schema SOURCE_PATH_GOAL, is 'going from Berlin to Prague'. Note that many cases do not provide information about START_PATH and END_PATH of a movement; e.g., 'leaving Berlin' and 'travelling to Berlin' are examples for the image schemas SOURCE_PATH and PATH_GOAL, respectively. 'Meandering' is an example of a concept that realises MOVEMENT_ALONG_PATH, which involves a PATH but no START_PATH or END_PATH. In contrast, no discernible PATH is involved in 'roaming the city', which is an example for MOVEMENT_OF_OBJECT. These examples illustrate that image schemas may be ordered hierarchically with respect to their content: SOURCE_PATH_GOAL contains more spatial primitives and more information than, for example, MOVEMENT_ALONG_PATH (the root of the PATH-following family), and MOVEMENT_ALONG_PATH, in turn, is more specific than MOVEMENT_OF_OBJECT, which in

fact lacks the spatial primitive 'path'. Figure 6 depicts the members and their connections involved in the PATH family. Beyond concepts that involve movement, PATH-following plays an important role in many abstract concepts and conceptual metaphors. For instance, the concept of 'going for a joy ride' realises the image schema SOURCE_PATH, since it has a START_PATH and a PATH but no END_PATH. Similarly, the expression 'running for president' describes the process of trying to get elected as president metaphorically as a PATH_GOAL. In this metaphor the PATH consists of the various stages of the process (e.g., announcing a candidacy and being nominated by a party) with the inauguration as END_PATH.

To implement computationally the idea of using image schemas as generic spaces, we need to deal with two independent algorithmic problems. First, we will have to identify an image schematic theory within the inputs. Then, if needed, we will have to find the least general (or, most specific) image schema common to both inputs, outlined in more detail in the next section.

Figure 7 shows a simple example of the underlying reasoning, using the well-known blend of the concepts *house* and *boat* into *houseboat* (or boathouse, or indeed many alternative combinations, see [36]). First, recognising image schemas in the axiomatisation of the concepts of *house* and *boat* comes down to identifying a few axioms in the input ontologies. Namely, boats host passengers or cargo, and travel from harbours to harbours. Thus there will be some axioms that axiomatise at least partly CONTAINMENT and SOURCE_PATH_GOAL. Further, houses are build on solid ground and host their inhabitants, so again we should be able to find axioms related to CONTAINMENT at least. To find such axioms a number of techniques have been used or can be further explored, including alingment techniques (axiomatisations will use different vocabularies for talking about containment) and machine learning techniques applied to text corpuses, see e.g., [21, 39]. Now, a common version of containment can be used as a base ontology, which will be inherited

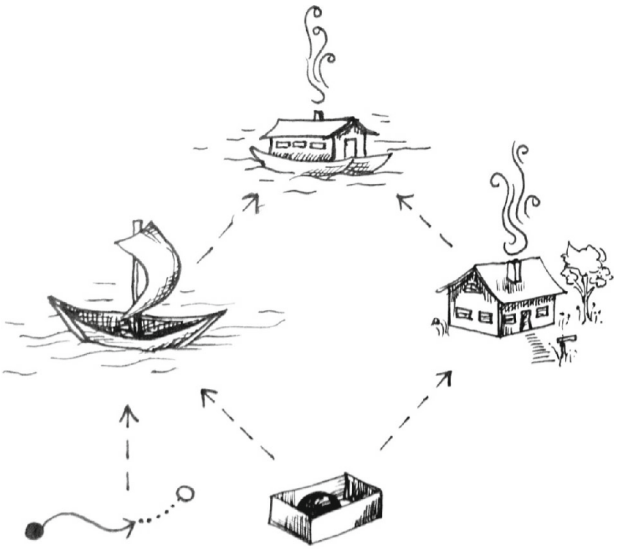

Fig. 7 Image schemas of SOURCE_PATH_GOAL and CONTAINMENT in the houseboat blend

to the blend. Such a base ontology for containment can be found, for instance, by refining both versions of containment to a common generalisation. Further, the particular blend of houseboat may inherit the SOURCE_PATH_GOAL of the boat to be a travelling houseboat, or it might be solidly grounded on water, further inheriting and mixing aspects of boats and houses, turning the created blend into a water-based residence. These choices are not determined by choosing the base ontology alone. Moreover, as the canonical house sits on solid ground, for the blend theory to be consistent, selection/forgetting, respectively debugging, will be necessary.

4.2.2 Recognising and generalising image schemas

We discussed above the idea that image schemas should be formally approached as interconnected families of theories, partially ordered by generality. This section demonstrates some of the benefits of using such a 'family structure' to represent image schemas, and how this added structure contributes to workflows for computational conceptual blending.

Figure 8 and the corresponding example given in Fig. 9 show the basic approach of using image schemas within the conceptual blending workflow. The central idea is that, when searching for ways to blend concepts, the image-schematic content that can be identified in the input ontologies takes priority in the definition of the base ontology over other information the input concepts might contain.

In Fig. 8, following the core model of blending described in Section 2.2, different image-schematic structures are first identified within the same image schema family in the two input concepts. They are then generalised to the most specific, common version within the image schema family to identify a generic space, using the pre-determined graph of image schemas (i.e., the least upper bound for two nodes in the family hierarchy is computed).

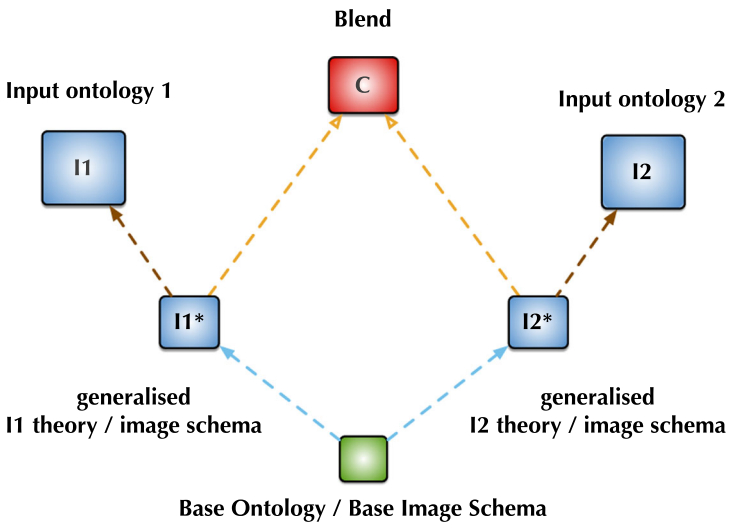

Fig. 8 Blending using common image schemas through theory generalisation

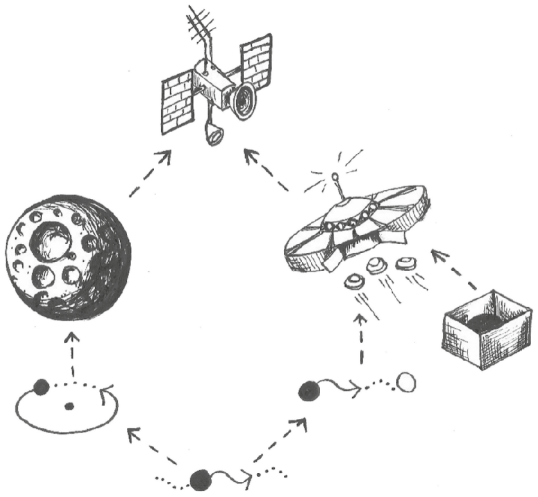

Fig. 9 The satellite as blended from moon and spaceship, and some of the involved image schemas

A slightly more complex situation is encountered in cases where we first specialise or complete the (description of) image schemas 'found' in the input concepts, before performing a generalisation step and finally identifying the generic space (below we will elaborate more on what it means to 'find' an image schema in an input space). This means moving down in the graph of the image schema family and choosing a member specified with more specificity. Of course, also a mix of these two basic approaches is reasonable, in other words, where the image schema of one input concept is specialised within a family whilst the other is generalised in order to identify a generic space based on joint image-schematic content.

To see these ideas at work, let us briefly discuss a more concrete example, taken from [44] where it is analysed in detail. The shown blend is an example of an iterative blend, taking as one of the input concepts the 'mothership' concept (itself a blend, i.e. the ontology defining the concept of a spaceship hosting smaller spaceships), and as the other input the concept of an orbiting moon. Clearly, both the generic spaceship as well as the orbiting moon have path-following image schemas build into their semantic setup, however arguably incompatible ones: whilst the orbit has no target but a 'circular' revolving movement (i.e. image schema REVOLVING_MOVEMENT), the SOURCE_PATH_GOAL of a spaceship has a target but no orbiting. To find the common core image schema, both need to be generalised, namely to the image schema SOURCE_PATH. This then allows, in the blend, to combine the movement patterns of the orbiting moon with aspects of the spaceship, to obtain a description of a satellite.

To implement computationally the idea of using image schemas as generic spaces, two independent algorithmic problems have to be solved. Namely:

(1) *Recognition Problem*: to identify an image-schematic theory within an input theory, and
(2) *Refinement Problem*: to find an appropriate image schema common to both inputs.

These tasks can be described in more detail as follows. To address the recognition problem, suppose a theory graph $\mathfrak{F}$ encoding an image schema family is fixed, like the example given in Fig. 6. For simplicity, it is assumed that elements of $\mathfrak{F}$ will be logical theories in a fixed formal logic, say description logic [49], image schema logic ISL [48], or standard first-order logic.[11]

Given an input theory I_1 and $\mathfrak{F}$, solving the recognition problem means finding a member $f \in \mathfrak{F}$ that can be *interpreted* in I_1, that is, such that a renaming σ of the symbols in f (called a signature morphism) is found and such that $I_1 \models \sigma(f)$ (also written $I_1 \models_\sigma f$).[12] Note that this is a more general statement than claiming the inclusion of the axioms of f (modulo renaming) in I_1 (the trivial inclusion interpretation) since establishing the entailment of the sentences in $\sigma(f)$ from I_1 might in fact be non-trivial, and the axioms needed for this quite different from the ones in f. In this context, notice that when iteratively applying generalisation refinement operators to a theory T, at each step, a generalisation T^* will be interpretable in the original theory T, and indeed, each theory S on the generalisation path between T^* and T will be thus interpretable.

Computational support for automatic theory-interpretation search in first-order logic is investigated in [71], and a prototypical system was developed and tested as an add-on to the Heterogeneous Tool Set (HETS) [67]. Experiments carried out in [54, 72] showed that this works particularly well with more complex axiomatisations in first-order logic, rather than with simple taxonomies expressed in, for instance, OWL. This is because, in the latter case too little syntactic structure is available to control the combinatorial explosion of the search task. From the point of view of interpreting image schemas into non-trivial axiomatised concepts, this can be seen as an encouraging fact, as image schemas are, despite their foundational nature, complex objects to axiomatise.

Once the recognition problem has been solved in principle, the given theory graph structure of the image schema family $\mathfrak{F}$ provides a simple handle on the generalisation problem. Namely, given two input spaces I_1, I_2, and two image schemas f_1, f_2 from the same family $\mathfrak{F}$ (say, 'CONTAINMENT') such that $I_1 \models_{\sigma_1} f_1$ and $I_2 \models_{\sigma_2} f_2$, compute the most specific generalisation $G \in \mathfrak{F}$ of f_1 and f_2, that is their least upper bound in $\mathfrak{F}$. Since the signature of G will be included in both signatures of f_1 and f_2, one obtains that $I_1 \models_{\sigma_1} G$ and $I_2 \models_{\sigma_2} G$. $G \in \mathfrak{F}$ is, therefore, an image schema common to both input spaces and can be used as generic space.

To address the generalisation and specialisation of image schema theories more generally, we can employ the notion of refinement operators introduced above. More specifically, the theories that are interpreted into the image schema theory graph can be upwards or downwards refined, in order to achieve an alternative image-schematic agreement between the two input spaces.

In order to implement these ideas more fully, a sufficiently comprehensive library of formalised image schemas, like the one presented above formalising the Path-related image schemas, needs to be made available for access by a blending engine.

In this line of thinking, a challenging research program would involve not to uniformly represent different kinds of image schemas in one expressive logic, such as first- or higher-order logic, but instead to construct a mapping between the cognitive levels of image schema

[11] Note that none of the ideas presented here depend on a particular fixed logic. Indeed, heterogeneous logical specification is central to formal blending approaches, see Kutz et al. [57].

[12] In more detail: a theory interpretation σ is a signature morphism renaming the symbols of the image schema theory f and induces a corresponding sentence translation map, also written σ, such that the translated sentences of f, written $\sigma(f)$, are logically entailed by I_1.

development and correspondingly adequate logical representations on different layers of abstraction. For instance, the gradual construction of an explicit representation of a timeline over which humans might meaningfully be able to quantify could be bootstrapped from the most basic path following image schema where a progression of time is completely implicit in the notion of movement along the path understood as a basic event. This line of work has been initiated in [44, 48].

5 Blend evaluation via axiom weakening

When using ontologies to formally represent concepts for blending, blended concepts can turn out to be logically inconsistent. A simple approach would consist in rejecting these blends, and to continue with the amalgam-based workflow to generate alternative, newly blended, and hopefully consistent, concepts [20, 24]. However, handling inconsistency in concept invention deserves attention and is often worthwhile since an inconsistent concept can be still a good blend once it is 'repaired'. The ontology engineering lifecycle, in general, includes a step for revision and debugging, just like a piece of software. Several ontology debugging techniques have been developed [8, 50, 51, 84], in particular in the Semantic Web community, and they could directly be adopted to repair inconsistent blends. The main drawback of these techniques is that, once they identified the problematic axioms, they remove them. Thus, they have the side effect of removing also many potentially wanted implicit consequences, and, from the concept invention point of view, potentially interesting characteristics of the blend.

Using alternative techniques, which allow for a more fine-grained method to ontology repair, is desirable. Refinement operators can be used to define axiom weakening, a technique that resolves inconsistencies by modifying axioms according to a background ontology [93], which we discuss next in greater detail.

5.1 Axiom weakening

Axiom weakening is a novel technique that can be employed to repair inconsistencies in logical theories, and in particular, the technique was developed for resolving inconsistencies in ontologies in the context of the DL $\mathcal{ALC}$ [93]. Compared to the more classical approach for repairing a description logic ontology, which amounts in deleting axioms, axiom weakening is a more gentle approach in which axioms are not deleted, but only weakened. A derivative line of work on the notion of axiom weakening established in [93] can be found in [10].

The general idea behind axiom weakening is that instead of removing axioms they are made more general. Intuitively, making an axiom of the form $C \sqsubseteq D$ more general amounts to enlarge the set of interpretations that satisfy the axiom. This can be achieved in different ways: either by substituting $C \sqsubseteq D$ with $C \sqsubseteq D'$, where D' is a more general concept than D; or, by modifying the axiom $C \sqsubseteq D$ to $C' \sqsubseteq D$, where C' is a more specific concept than C; or even by generalising and specialising simultaneously to obtain $C' \sqsubseteq D'$.

Axiom weakening is defined by means of generalisation and specialisation refinement operators. In the following, we note nnf the function that for every concept C, returns its negation normal form $\mathsf{nnf}(C)$. The negation normal form of an $\mathcal{ALC}$ concept or formula is a formula in which negation appears only in the front of atomic concepts. It is possible to transform any $\mathcal{ALC}$ formula into an equivalent one in negation normal form.

Definition 12 Let $\mathcal{T}$be an $\mathcal{ALC}$ TBox. We define $\gamma_{\mathcal{T}}$, the *generalisation refinement operator* w.r.t. $\mathcal{T}$, inductively over the structure of concept descriptions as:

$$\begin{aligned}
\gamma_{\mathcal{T}}(A) &= \mathsf{UpCov}_{\mathcal{T}}(A)\\
\gamma_{\mathcal{T}}(\neg A) &= \{\mathsf{nnf}(\neg C) \mid C \in \mathsf{DownCov}_{\mathcal{T}}(A)\} \cup \mathsf{UpCov}_{\mathcal{T}}(\neg A)\\
\gamma_{\mathcal{T}}(\top) &= \mathsf{UpCov}_{\mathcal{T}}(\top)\\
\gamma_{\mathcal{T}}(\bot) &= \mathsf{UpCov}_{\mathcal{T}}(\bot)\\
\gamma_{\mathcal{T}}(C \sqcap D) &= \{C' \sqcap D \mid C' \in \gamma_{\mathcal{T}}(C)\} \cup \{C \sqcap D' \mid D' \in \gamma_{\mathcal{T}}(D)\} \cup \mathsf{UpCov}_{\mathcal{T}}(C \sqcap D)\\
\gamma_{\mathcal{T}}(C \sqcup D) &= \{C' \sqcup D \mid C' \in \gamma_{\mathcal{T}}(C)\} \cup \{C \sqcup D' \mid D' \in \gamma_{\mathcal{T}}(D)\} \cup \mathsf{UpCov}_{\mathcal{T}}(C \sqcup D)\\
\gamma_{\mathcal{T}}(\forall R.C) &= \{\forall R.C' \mid C' \in \gamma_{\mathcal{T}}(C)\} \cup \mathsf{UpCov}_{\mathcal{T}}(\forall R.C)\\
\gamma_{\mathcal{T}}(\exists R.C) &= \{\exists R.C' \mid C' \in \gamma_{\mathcal{T}}(C)\} \cup \mathsf{UpCov}_{\mathcal{T}}(\exists R.C)
\end{aligned}$$

Definition 13 Let $\mathcal{T}$be an $\mathcal{ALC}$ TBox. We define $\rho_{\mathcal{T}}$, the *specialisation refinement operator* w.r.t. $\mathcal{T}$, inductively over the structure of concept descriptions as:

$$\begin{aligned}
\rho_{\mathcal{T}}(A) &= \mathsf{DownCov}_{\mathcal{T}}(A)\\
\rho_{\mathcal{T}}(\neg A) &= \{\mathsf{nnf}(\neg C) \mid C \in \mathsf{UpCov}_{\mathcal{T}}(A)\} \cup \mathsf{DownCov}_{\mathcal{T}}(\neg A)\\
\rho_{\mathcal{T}}(\top) &= \mathsf{DownCov}_{\mathcal{T}}(\top)\\
\rho_{\mathcal{T}}(\bot) &= \mathsf{DownCov}_{\mathcal{T}}(\bot)\\
\rho_{\mathcal{T}}(C \sqcap D) &= \{C' \sqcap D \mid C' \in \rho\mathcal{T}(C)\} \cup \{C \sqcap D' \mid D' \in \rho_{\mathcal{T}}(D)\} \cup \mathsf{DownCov}_{\mathcal{T}}(C \sqcap D)\\
\rho_{\mathcal{T}}(C \sqcup D) &= \{C' \sqcup D \mid C' \in \rho\mathcal{T}(C)\} \cup \{C \sqcup D' \mid D' \in \rho_{\mathcal{T}}(D)\} \cup \mathsf{DownCov}_{\mathcal{T}}(C \sqcup D)\\
\rho_{\mathcal{T}}(\forall R.C) &= \{\forall R.C' \mid C' \in \rho_{\mathcal{T}}(C)\} \cup \mathsf{DownCov}_{\mathcal{T}}(\forall R.C)\\
\gamma_{\mathcal{T}}(\exists R.C) &= \{\exists R.C' \mid C' \in \rho_{\mathcal{T}}(C)\} \cup \mathsf{DownCov}_{\mathcal{T}}(\exists R.C)
\end{aligned}$$

The $\mathsf{DownCov}_{\mathcal{T}}$ set of a concept C is the set of most general descriptions that are subsumed by C (a similar definition to the one already provided for $\mathsf{UpCov}_{\mathcal{T}}$ in Section 4.1).

The refinement operators can be used as components of a method for repairing inconsistent ontologies by weakening, instead of removing, problematic axioms. However, given an inconsistent ontology O, the upcover and downcover are trivially defined. To this end, we first need to find a consistent subontology $\hat{O}$ of O to serve as *reference ontology*.

Different options can be explored: one can pick a random maximally consistent subset of O as a reference ontology $\hat{O}$ (*brave* approach). On the contrary, one can take as $\hat{O}$ the intersection of all maximally consistent subsets (*cautious* approach). While the brave approach is faster to compute and still guarantees to find solutions, the cautious approach has the advantage of not excluding certain repairs a priori.[13]

Given a reference ontology $\hat{O}$, axiom weakening is defined as follows:

Definition 14 (**Axiom weakening**) Given a subsumption axiom $C \sqsubseteq D$ in an ontology O, the set of (least) *weakenings* of $C \sqsubseteq D$ w.r.t. $\hat{O}$, denoted by $g_{\hat{O}}(C \sqsubseteq D)$ is the set of all axioms $C' \sqsubseteq D'$ such that:

$$\begin{aligned}
&C' \in \rho_{\hat{O}}(C) \text{ and } D = D' \text{ or}\\
&D' \in \gamma_{\hat{O}}(D) \text{ and } C' = C \ .
\end{aligned}$$

[13] For a further discussion about how to choose a reference ontology and its implication to axiom weakening we refer to [80, 81].

Algorithm 1 Repairing blends through weakening.

Procedure FIX-BLEND(B, $\hat{O}$) ▷ B inconsistent blend, $\hat{O}$ reference ontology

1: **while** B is inconsistent **do**
2: $\psi \leftarrow$ FindBadAxiom(B) ▷ find bad axiom occurring often in MCSs of B
3: **choose** $\psi' \in g_{\hat{O}}(\psi)$
4: $B \leftarrow (B \setminus \{\psi\}) \cup \{\psi'\}$
5: **return** B

Given an assertional axiom $C(a)$ of O, the set of (least) *weakenings* of $C(a)$, denoted $g_{\hat{O}}(C(a))$ is the set of all axioms $C'(a)$ such that

$$C' \in \gamma_{\hat{O}}(C) \ .$$

It is important to point out that given a consistent reference ontology $\hat{O}$, the weakening of an axiom is always satisfied by a superset of the interpretations that satisfy the axiom. Therefore for every subsumption or assertional axiom ϕ, the axioms in the set $g_{\hat{O}}(\phi)$ are indeed weaker than ϕ. Technical details can be found in [93].

5.2 Repairing blends

Any inconsistent set of axioms Y can in principle be repaired by means of a sequence of weakenings of the axioms in Y with respect to $\hat{O}$. Thus, inconsistent blends can also be repaired. In the context of blend evaluation, minimal inconsistent subsets (MCSs) are very interesting since they represent alternatives of how a blend can be repaired, and they can suggest how to revise it in a minimal way. This information is useful for deciding which axioms to weaken whilst to keep as much information as possible in the blend description.

When a blend B (an ontology) is inconsistent, we can adopt the strategy described in Algorithm 1 to repair it w.r.t. a given (fixed) reference ontology $\hat{O}$. The algorithm finds the minimally inconsistent subsets of B (e.g., using the methods from [5, 84]) and finds the most problematic axioms, namely, axioms that appear more often in the MCSs of B (FindBaxAxiom). Then, one of these axioms is repaired by adding a weakened version to the ontology. The process is repeated until a consistent ontology is found.

Ontology	Input Concepts	Blended Concept
1. House $\sqsubseteq$ Object		
2. Boat $\sqsubseteq$ Object		
3. Land $\sqsubseteq$ Medium	11. House $\sqsubseteq \forall$usedBy.Resident	
4. Water $\sqsubseteq$ Medium	12. House $\sqsubseteq \forall$on.Land	
5. Resident $\sqsubseteq$ Person		15. Blend $\sqsubseteq$ House $\sqcap$ Boat
6. Passenger $\sqsubseteq$ Person		16. Blend(HouseBoat)
7. Water $\sqcap$ Land $\sqsubseteq \bot$	13. Boat $\sqsubseteq \forall$usedBy.Passenger	
8. Person $\sqcap$ Medium $\sqsubseteq \bot$	14. Boat $\sqsubseteq \forall$on.Water	
9. Object $\sqcap$ Medium $\sqsubseteq \bot$		
10. Object $\sqcap$ Person $\sqsubseteq \bot$		

Fig. 10 The ontology of the houseboat concept: the left of the table contains the common knowledge (axioms 1-10) used by the input concept House (axioms 11-12) and Boat (axioms 13-14). The right side of the table contains the blended concept, which in this example is defined as the conjunction of the House and the Boat concepts. The ontology is inconsistent due to the disjointness axiom 7 and the assertional axiom 16

To illustrate an example of blend repairing, we use the classical conceptual blending example of the *houseboat* [27, 33]. In Fig. 10, we depict the set of axioms, describing the houseboat blend. The precise formalisation is not critical at this point, different ones exist [35, 79], but all provide similar distinctions.

The axioms in the blend denote that a house is an object that is only used by residents and is located only on land. Similarly, boat is an object that is only used by passengers and is located only on water; and, a house-boat is a combination of a house and a boat.

The HouseBoat theory is inconsistent since we can infer that Blend is an object located both on land and water, but this is 'disallowed' due to the disjointness axiom-7 ($\mathsf{Water} \sqcap \mathsf{Land} \sqsubseteq \bot$). Since the Blend concept is unsatisfiable and it is instantiated by the assertion axiom-16, the ontology B is inconsistent. The blend, however, is still to be considered an interesting option from a creative point of view. For instance, by repairing the HouseBoat ontology it would be possible to 'create' the concept of a 'floating house' (thus a house on water), or the concept of a 'movable house on land', among others.

Algorithm 1 can be applied as follow:

- We select the maximally consistent subset $B \setminus \{7\}$ as a reference ontology $\hat{O}$.
- Out of the axioms in B, the algorithm then could randomly choose between the axioms-7, 11, 12, 13, 14 and 15, which are the "bad" axioms responsible for the inconsistency. Let us say it picks axiom-12 $\mathsf{House} \sqsubseteq \forall \mathsf{on.Land}$.
- Among the weakenings of axiom-12, there is $\mathsf{House} \sqsubseteq \forall \mathsf{on.Medium}$ which is used to replace axiom 12.
- B now is consistent (Fig. 11), and the repairing mechanism is over. The consistent blend theory obtained captures the idea of a 'floating house'.

Clearly, there exist different possible repairs that depend both on the reference ontology selected and on the weakenings that are used to solve the inconsistency. E.g., alternative reference ontologies that could be adopted are: $B \setminus \{16\}$, $B \setminus \{12\}$, or $B \setminus \{14\}$; other possible blend repairs could be obtained by weakening axiom-14 into $\bot \sqsubseteq \forall \mathsf{on.Water}$, or axiom-15 into $\mathsf{Blend} \sqsubseteq \mathsf{Boat} \sqcap \forall \mathsf{usedBy.Resident}$, or axiom-12 into $\mathsf{Boat} \sqcap \mathsf{House} \sqsubseteq \forall \mathsf{on.Land}$ and then into $\mathsf{Boat} \sqcap \mathsf{House} \sqsubseteq \forall \mathsf{on.Medium}$, etc.

From a conceptual blending point of view, not all the repaired blends are interesting, and the mechanism of ontology repair should be combined with other selection criteria or heuristics. This could be done, for instance, by considering (ontological) consequence requirements, i.e., a collection of conjectures that represent desired properties that a blend

Ontology	Input Concepts	Blended Concept
1. $\mathsf{House} \sqsubseteq \mathsf{Object}$		
2. $\mathsf{Boat} \sqsubseteq \mathsf{Object}$		
3. $\mathsf{Land} \sqsubseteq \mathsf{Medium}$	11. $\mathsf{House} \sqsubseteq \forall \mathsf{usedBy.Resident}$	
4. $\mathsf{Water} \sqsubseteq \mathsf{Medium}$	12w. $\mathsf{House} \sqsubseteq \forall \mathsf{on.Medium}$	
5. $\mathsf{Resident} \sqsubseteq \mathsf{Person}$		15. $\mathsf{Blend} \sqsubseteq \mathsf{House} \sqcap \mathsf{Boat}$
6. $\mathsf{Passenger} \sqsubseteq \mathsf{Person}$		16. $\mathsf{Blend}(\mathsf{HouseBoat})$
7. $\mathsf{Water} \sqcap \mathsf{Land} \sqsubseteq \bot$	13. $\mathsf{Boat} \sqsubseteq \forall \mathsf{usedBy.Passenger}$	
8. $\mathsf{Person} \sqcap \mathsf{Medium} \sqsubseteq \bot$	14. $\mathsf{Boat} \sqsubseteq \forall \mathsf{on.Water}$	
9. $\mathsf{Object} \sqcap \mathsf{Medium} \sqsubseteq \bot$		
10. $\mathsf{Object} \sqcap \mathsf{Person} \sqsubseteq \bot$		

Fig. 11 The repaired ontology of the HouseBoat Blend obtained by weakening $\mathsf{House} \sqsubseteq \forall \mathsf{on.Land}$ (axiom 12) into $\mathsf{House} \sqsubseteq \forall \mathsf{on.Medium}$ (axiom 12w)

should have. Satisfaction of these requirements can then be used as a criterion to decide whether to accept or reject a repaired blend. If the repaired blend is 'rejected' according to this criterion, then different weakenings can be used. Ideally, the results of the evaluation are also supposed to guide the changes in the next cycle of the blend workflow, namely, to choose the generalisations and the generic space to be used in the amalgams.

Besides, concept invention can be naturally conceived as a collaborative creative process. In particular, the role of inconsistency in concept discovery and invention was explored by Lakatos [62]. Lakatos was a philosopher of mathematics who developed a model of argumentation of 'proofs' and 'refutations' to describe ways in which mathematicians explore and develop new concepts. In particular, he looked at the role that conflict plays in such explorations, presenting a rational reconstruction of a dialogue in which claims are made and counterexamples are presented and responded to in various different ways. Lakatosian reasoning is an ideal example of how a concept can be repaired, by pinpointing axioms, creating and exchanging arguments over the axioms, and deciding how these cohere with other axioms [78].

Therefore, the Lakatosian way of conceiving concept evaluation, perhaps best described as a form of concept evolution, is a very useful guiding paradigm for collective blend repair. Providing a full computational account for such a concept evolution framework driven by argumentation remains a desirable goal for future research.

Towards this, an approach inspired by previous work in computational social choice for collective blend repair was conceived in [80, 81]. This is covered in the next section.

5.3 Repairing blends collectively

The idea behind repairing a blend in a collective way is that different experts having different preferences and opinions over the specifics of an (inconsistent) blend engage in a turn-based repairing mechanism to decide i) which characteristics the repaired blend should have, and ii) how to repair the blend [81]. This collective mechanism is of particular importance when one wants to incorporate social aspects into the creation process, such as for instance the selection of certain blends' specifics depending on personal tastes or different socio-demographic contexts. Other examples in which 'collectivity' (in concept invention) is important are the re-construction of the intended meaning of a blend (see e.g. [25]), or the refinement of the intended meaning in the sense of 'running of the blend' as discussed in the orignal formulation of conceptual blending [26], i.e. where missing information is added by a human 'interpreter' (see e.g. [17]).

In this setting, experts can be modeled as agents who share an inconsistent set of statements about a blended theory (expressed as axioms in a description logic). Each expert i submits a (consistent) subset of axioms O_i of the blend, and a preference profile $<_i$ over these axioms, which is a total ordering that reflects the agents' view of the importance of the statements for the description of the blend.

Then, the experts arbitrarily take turns adding their 'favorite' axiom to a set of previously selected axioms. When their favorite axiom cannot be added without causing an inconsistency, this axiom is weakened using a reference ontology $\hat{O}$. The reference ontology can be obtained according to brave or cautious criteria (Section 5.1), or even according to criteria based on the preferences profiles of the agents [81]. The procedure ends when all the axioms of the blend that are supported by at least one expert have been considered (and so added as such or in a weakened form).

This procedure, described in Algorithm 2, works as follows:

Algorithm 2 CollectiveBlendRepair(B, $\hat{O}$, $(<_i)_i$, $(O_i)_i$)

$\hat{B} \leftarrow \emptyset$
TreatedAxioms $\leftarrow \emptyset$
Agent $\leftarrow 1$
while not all agents have finished **do**
  if every axiom in O_{Agent} is treated **then**
    Agent has finished
  else
    Ax $\leftarrow$ FavoriteUntreatedAxiom($<_{\text{Agent}}$, O_{Agent})
    SetToTreated(Ax)
    while $\hat{B} \cup \{Ax\}$ is inconsistent **do**
      Ax $\leftarrow$ WeakenAxiom(Ax, $\hat{O}$)
    $\hat{B} \leftarrow \hat{B} \cup \{Ax\}$
  Agent $\leftarrow$ (Agent mod $|\mathcal{A}|) + 1$
Return $\hat{B}$

1. Compute a reference ontology $\hat{O}$ of the inconsistent blend B. Set the repaired blend $\hat{B}$ to empty.
2. By turn, each agent i considers their next preferred axiom in their set O_i of chosen axioms.
3. If an agent i does not have any more axioms to propose (when TreatedAxiom$\cap O_i = \emptyset$), then it skips. Agent i has finished.
4. Otherwise, agent i picks FavoriteUntreatedAxiom($<_i$, O_i), which is its favorite axiom Ax in the set $(B \setminus \text{TreatedAxioms}) \cap O_i$. Then, as long as $\hat{B} \cup \{Ax\}$ is inconsistent, it weakens it w.r.t. the reference ontology: Ax is set to one of its weakenings. $\hat{B}$ is then set to $\hat{B} \cup \{Ax\}$.
5. Agents repeat steps 2–4 until they have processed all their chosen axioms.

Returning to the ontology B for the HouseBoat blend from Fig. 10, suppose that three experts submit their opinions on B as shown next:

Expert 1	$<_1 = 6 < 2 < 3 < 4 < 15 < 8 < 10 < 13 < 9 < 14 < 7 < 1 < 16 < 5 < 12 < 11$
	$O_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16\}$
Expert 2	$<_2 = 6 < 10 < 11 < 12 < 16 < 13 < 4 < 15 < 9 < 5 < 8 < 2 < 14 < 3 < 1 < 7$
	$O_2 = \{2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$
Expert 3	$<_3 = 16 < 9 < 7 < 13 < 10 < 15 < 6 < 4 < 2 < 11 < 14 < 3 < 8 < 5 < 1 < 12$
	$O_3 = \{2, 4, 6, 7, 9, 10, 11, 13, 15, 16\}$

Observe that the opinions of Experts 1, 2, and 3 reflect different points of view about the blend. They all agree on the assertion and on the blend concept, but not on some of its specifics. For instance, Expert-1 does not select the specific that a boat should be used by a passenger, whereas for Experts 2 and 3 this specific matters. Furthermore, Experts 1 and 2 consider the specific of a house being on land relevant.

The turn-based mechanism proceeds as follows. As reference ontology $\hat{O}$ the experts consider $B \setminus \{16\}$, that is, the blend ontology minus the assertion axiom axiom 16 (which causes the ontology to be inconsistent). We initialize $\hat{B} = \emptyset$.

1. Round
   - Expert 1 chooses axiom 6. $\hat{B} = \{6\}$.
   - Expert 2 chooses axiom 10. The favorite choice, axiom 6, has already been treated. $\hat{B} = \{6, 10\}$.
   - Expert 3 chooses axiom 16. $\hat{B} = \{6, 10, 16\}$.
2. Round
   - Expert 1 chooses axiom 2. $\hat{B} = \{2, 6, 10, 16\}$.
   - Expert 2 chooses axiom 11. The favorite choice, axiom 10, has already been treated. $\hat{B} = \{2, 6, 10, 11, 16\}$.
   - Expert 3 chooses axiom 9. $\hat{B} = \{2, 6, 9, 10, 11, 16\}$.
3. Round
   - Expert 1 chooses axiom 3. $\hat{B} = \{2, 3, 6, 9, 10, 11, 16\}$.
   - Expert 2 chooses axiom 12. $\hat{B} = \{2, 3, 6, 9, 10, 11, 12, 16\}$.
   - Expert 3 chooses axiom 7. $\hat{B} = \{2, 3, 6, 7, 9, 10, 11, 12, 16\}$.
4. Round
   - Expert 1 chooses axiom 4. $\hat{B} = \{2, 3, 4, 6, 7, 9, 10, 11, 12, 16\}$.
   - Expert 2 chooses axiom 13. The favorite choice, axiom 16, has already been treated. $\hat{B} = \{2, 3, 4, 6, 7, 9, 10, 11, 12, 13, 16\}$.
   - Expert 3 chooses axiom 15. The favorite choices, axioms 13 and 10, have already been treated. $\hat{B} = \{2, 3, 4, 6, 7, 9, 10, 11, 12, 13, 15, 16\}$.
5. Round
   - Expert 1 chooses axiom 8. The favorite choice, axiom 15, has already been treated. $\hat{B} = \{2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16\}$.
   - Expert 2 chooses axiom 5. The favorite choices, axioms 4, 15 and 9, have already been treated. $\hat{B} = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16\}$.
   - Expert 3 does not have any other axioms to propose.
6. Round
   - Expert 1 chooses axiom 14 ($\mathsf{Boat} \sqsubseteq \forall \mathsf{on}.\mathsf{Water}$). The favorite choices, axioms 10 and 9, have already been treated. However, axiom 14 is inconsistent with the axioms that are already present in $\hat{B}$. It is then weakened into

     $$(\mathsf{Boat} \sqsubseteq \forall \mathsf{on}.\mathsf{Medium}),$$

     that we denote as axiom $14w$. $\hat{B} = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14w, 15, 16\}$.
   - Expert 2 does not have any other axioms to propose.
   - Expert 3 skips.
7. Round
   - Expert 1 chooses axiom 1. $\hat{B} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14w, 15, 16\}$.
   - Expert 2 skips.
   - Expert 3 skips.

The ontology $\hat{B}$ resulting from this mechanism is very similar to the blend ontology B, but axiom $14w$ was weakened. The resulting ontology, which is consistent, captures the idea of the boat-house blend, which is a 'movable house'.

The termination of CollectiveBlendRepair is easy to see: at each step, an agent finishes, or an axiom from B is set as treated. Clearly, substituting an axiom ψ with an element from $g_{\hat{O}}(\psi)$ cannot diminish the set of models of an ontology, and any axiom is a finite number of refinement steps away from the trivial axiom $\bot \sqsubseteq \top$ [81]. Any assertion $C(a)$ is also a finite number of generalisations away from the trivial assertion $\top(a)$. It follows that by repeatedly replacing an axiom with one of its weakenings, the weakening procedure will eventually obtain an ontology with some interpretations. Hence, the algorithm terminates.

6 Discussion, conclusion and future perspectives

In this paper we presented a core model of computational conceptual blending and several techniques, tools and frameworks, that support its realisation. We discussed two open problems in the implementation of a computational framework for conceptual blending: the search for a generic space, and the (logical) evaluation of the blends. We presented different solutions for these problems from the perspective of refinement operators.

Refinement operators are used to specify a refinement space according to which concept descriptions can be made more general or more specific. We showed how refinement operators play an important role in the search for a generic space, and in the blend evaluation. Indeed, they support the generalisation of input concepts, the refinement of image schemas, and the definition of axiom weakening for repairing inconsistent blends.

As far as the generic space search is concerned, although refinement operators are an effective way to look for the commonalities between input spaces, typically for any input concepts the space of generalisations is quite large and exploring this space can be quite costly in terms of computation time. A mechanism to 'restrict' the search space is desirable. To this end, image schemas can provide a means to distinguish cognitively relevant from irrelevant information.

Image schemas are cognitive structures that define knowledge patterns encapsulating objects' properties and affordances. Identifying image schemas in the input spaces to be blended can improve the focus of the blending, thus limiting the space of the generalisations to be explored. Whilst from a formal perspective some advances have already been made, more attention still needs to be paid to building a more unified terminology integrating the formal and cognitive-linguistic approaches found in the literature [43]. Similar studies have been carried out in the area of music cognition [2, 3, 15]. Besides, image schemas should be systematically formalised as families of logical theories, before being able to effectively use them to steer the generic space search. A closely related approach for steering the computation of a generic space is to consider shared (resp. non-shared) ontology patterns and anti-patterns [28, 42, 59, 90]. Here, libraries of ontology patterns can play a similar role to image schemas, as they capture schematically common ontological structure, and part of the generic space computation can be turned into an ontology-patterns matching problem. Similarly, anti-patterns, understood as schematically captured modelling situations that should typically be avoided, can serve an important role in blend evaluation.

From the blend evaluation point of view more generally, we showed how inconsistent blends can be repaired by weakening some axioms. However, by using ontology debugging in order to detect and solve inconsistencies, the number of different repairs may be huge

(exponential) and therefore difficult to navigate. In the context of an AI application as for instance conceptual blending, this challenge is amplified by the fact that in the search for new 'ideas', the terms, concepts, or ontologies to be combined or blended are selected based on rich semantic clashes, following the intuition of conceptual blending. This means that one needs, in principle, to explore many repairs to identify those that can be interesting.

In this respect, an ontology debugging technique, such as axiom weakening, should be combined with other techniques to foster better blend evaluation abilities in a computational framework of concept invention. Previous research in this direction was initiated in [19, 87], where a first formal account for the conceptual coherence theory of Thagard [92] was proposed and its relationship w.r.t. semantic similarity was explored. Conceptual coherence can provide a mechanism to find the degree of relatedness between elements in an ontology, to identify the most coherent, although perhaps still inconsistent, axioms to weaken. A closely related approach is that of weighted concept definitions, motivated by prototype theory, where the individual contributions of facets of a concept's definition are accumulated in oder to be classified under a concept [82]. Adopting a more holistic approach based on axiom weakening, coherence and similarity, and more generally cognitively inspired models of concept formation, should provide a more effective means to implement blend evaluation, and it is an interesting approach to be explored.

Acknowledgements Work on this paper was supported by the project 'Computational Technologies for Concept Invention' (COCO) funded by the Province of Bozen-Bolzano–South Tyrol.

The paper builds on a body of work originating from the EU FP7 project COINVENT and later refinements. We would like to thank our co-authors of prior work, in alphabetical order: Tarek R. Besold, Manfred Eppe, Pietro Galliani, Maria M. Hedblom, Till Mossakowski, Fabian Neuhaus, Rafael Peñaloza, Enric Plaza, Daniele Porello, Marco Schorlemmer, and Nicolas Troquard.

We also sincerely thank the anonymous reviewers for their valuable and detailed feedback that helped improve this paper.

This work was supported by the Open Access Publishing Fund provided by the Free University of Bozen-Bolzano.

References

1. Anderson, A.R., Belnap, N.D., Dunn, J.: Entailment: The Logic of Relevance and Necessity, vol. 2. Princeton University Press, Princeton (1992)
2. Antovic, M.: Musical metaphors in Serbian and Romani children: An empirical study. Metaphor. Symb. **24**(3), 184–202 (2009)
3. Antovic, M., Bennett, A., Turner, M.: Running in circles or moving along lines: Conceptualization of musical elements in sighted and blind children. Music. Sci. **17**(2), 229–245 (2013). https://doi.org/10.1177/1029864913481470
4. Astesiano, E., Bidoit, M., Krieg-Brückner, B., Mosses, P.D., Sannella, D., Tarlecki, A.: CASL: The common algebraic specification language. Theor. Comput. Sci. **286**(2), 153–196 (2002)
5. Baader, F., Peñaloza, R.: Axiom pinpointing in general tableaux. J. Logic Comput. **20**(1), 5–34 (2010). special Issue: Tableaux and Analytic Proof Methods
6. Baader, F., Calvanese, D., Mcguinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook. Cambridge University Press (2003)
7. Baader, F., Brandt, S., Lutz, C.: Pushing the EL Envelope. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence, pp. 364-*369. Morgan Kaufmann Publishers Inc., San Francisco (2005)

8. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic $\mathcal{EL}^+$. In: Proc. of KI 2007, vol. 4667, pp. 52–67. Springer, LNCS (2007)
9. Baader, F., Sertkaya, B., Turhan, A.Y.: Computing the least common subsumer w.r.t. a background terminology. J. Appl. Log. **5**(3), 392–420 (2007)
10. Baader, F., Kriegel, F., Nuradiansyah, A., Peñaloza, R.: Making repairs in description logics more gentle. In: KR, pp. 319–328. AAAI Press (2018)
11. Baumgartner, P., Fuchs, A., Tinelli, C.: Darwin: A theorem prover for the model evolution calculus. In: Schulz, S., Sutcliffe, G., Tammet, T. (eds.) IJCAR Workshop on Empirically Successful First Order Reasoning (ESFor (aka S4)). Electronic Notes in Theoretical Computer Science (2004)
12. Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. J. Log. Comput. **13**(3), 429–448 (2003)
13. Boden, M.A.: The Creative Mind: Myths and Mechanisms. George Weidenfeld and Nicolson Ltd (1990)
14. Bou, F., Eppe, M., Plaza, E., Schorlemmer, M.: D2.1: Reasoning with Amalgams. Tech. rep., COINVENT Project, available at http://www.coinvent-project.eu/fileadmin/publications/D2.1.pdf (2014)
15. Cambouropoulos, E., Kaliakatsos-Papakostas, M., Kühnberger, K.U., Kutz, O., ASmaill: Concept invention and music: Creating novel harmonies via conceptual blending. In: Proc.of the 9th Int.Conference on Interdisciplinary Musicology (CIM-2014). Berlin (2014)
16. Colton, S., Wiggins, G.: Computational creativity: the final frontier? In: ECAI, vol. 12, pp. 21–26 (2012)
17. Confalonieri, R., Corneli, J., Pease, A., Plaza, E., Schorlemmer, M.: Using argumentation to evaluate concept blends in combinatorial creativity. In: Proceedings of the 6th International Conference on Computational Creativity, ICCC15 (2015)
18. Confalonieri, R., Plaza, E., Schorlemmer, M.: A process model for concept invention. In: International Conference on Computational Creativity (ICCC16). Paris, France, 1st July (2016)
19. Confalonieri, R., Kutz, O., Galliani, P., Peñaloza, R., Porello, D., Schorlemmer, M., Troquard, N.: Coherence, similarity, and concept generalisation. In: Description Logics, CEUR-WS.org, CEUR Workshop Proceedings, vol. 1879 (2017)
20. Confalonieri, R., Eppe, M., Schorlemmer, M., Kutz, O., Peñaloza, R., Plaza, E.: Upward refinement operators for conceptual blending in the description logic $\mathcal{EL}^{++}$. Ann. Math. Artif. Intell. **82**(1), 69–99 (2018)
21. Confalonieri, R., Pease, A., Schorlemmer, M., Besold, T.R., Kutz, O., Maclean, E., Kaliakatsos-Papakostas, M.: Concept Invention: Foundations, Implementation, Social Aspects and Applications 1st edn. Computational Synthesis and Creative Systems. Springer International Publishing (2018)
22. Eppe, M., Confalonieri, R., Maclean, E., Kaliakatsos-Papakostas, M.A., Cambouropoulos, E., Schorlemmer, W.M., Codescu, M., Kühnberger, K.: Computational invention of cadences and chord progressions by conceptual chord-blending. In: Yang, Q., Wooldridge, M. (eds.) Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, pp. 2445–2451. AAAI Press (2015)
23. Eppe, M., Maclean, E., Confalonieri, R., Kutz, O., Schorlemmer, W.M., Plaza, E.: Asp, amalgamation, and the conceptual blending workflow. In: Calimeri, F., Ianni, G., Truszczynski, M. (eds.) Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings, vol. 9345, pp. 309–316. Springer, Lecture Notes in Computer Science (2015)
24. Eppe, M., Maclean, E., Confalonieri, R., Kutz, O., Schorlemmer, M., Plaza, E., Kühnberger, K.U.: A computational framework for conceptual blending. Artif. Intell. **256**, 105–129 (2018)
25. Falomir, Z., Plaza, E.: Towards a model of creative understanding: Deconstructing and recreating conceptual blends using image schemas and qualitative spatial descriptors. Annals of Mathematics and Artificial Intelligence. https://doi.org/10.1007/s10472-019-09619-9 (2019)
26. Fauconnier, G., Turner, M.: Conceptual integration networks. Cognit. Sci. **22**(2), 133–187 (1998)
27. Fauconnier, G., Turner, M.: The Way We Think: Conceptual Blending and the Mind's Hidden Complexities. Basic Books (2003)
28. Gangemi, A., Presutti, V.: Ontology design patterns. In: Handbook on Ontologies. International Handbooks on Information Systems, pp. 221–243. Springer (2009)
29. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Clingo = ASP + control: Preliminary report. arXiv:abs/1405.3694 (2014)
30. Gelfond, M., Kahl, Y.: Knowledge Representation, Reasoning, and the Design of Intelligent Agents: The Answer-Set Programming Approach. Cambridge University Press, New York (2014)
31. Gentner, D.: Why we're so smart? In: Gentner, D., Goldin-Meadow, S. (eds.) Language in Mind: Advances in the Study of Language and Thought, pp. 195–235. MIT Press (2003). chap 8

32. Goguen, J.: What is a concept? In: Dau, F., Mugnier, M.L., Stumme, G. (eds.) Conceptual Structures: Common Semantics for Sharing Knowledge. 13th International Conference on Conceptual Structures, ICCS 2005, Kassel, Germany, July 17-22, 2005. Proceedings, vol. 3596, pp. 52–77. Springer, Lecture Notes in Artificial Intelligence (2005)
33. Goguen, J.A.: An introduction to algebraic semiotics, with applications to user interface design. In: Computation for Metaphors, Analogy and Agents, no. 1562 in LNCS, pp. 242–291. Springer (1999)
34. Goguen, J.A.: Semiotic morphisms, representations and blending for interface design. In: Proc.of the AMAST Workshop on Algebraic Methods in Language Processing, pp. 1–15. AMAST Press (2003)
35. Goguen, J.A., Harrell, D.F.: Style: A computational and conceptual blending-based approach. In: Argamon, S., Dubnov, S. (eds.) The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning, pp. 147–170. Springer, Berlin (2010)
36. Goguen, J.A., Harrell, D.F.: (Last accessed, February 2019) Foundations for active multimedia narrative: Semiotic spaces and structural blending. Available at https://csewebucsdedu/goguen/pps/narrpdf
37. Goguen, J.A., Malcolm, G.: Algebraic Semantics of Imperative Programs. MIT (1996)
38. Grice, H.P.: Logic and conversation. In: Cole, P., Morgan, J.L. (eds.) Syntax and Semantics: Speech Acts, vol. 3, pp. 41–58. Academic Press, New York (1975)
39. Gromann, D., Hedblom, M.M.: Kinesthetic mind reader: A method to identify image schemas in natural language. In: Langley, P. (ed.) Proc.of the 5th Annual Conference on Advances in Cogntive Systems (ACS). Troy, New York (2017)
40. Grüninger, M., Fox, M.S.: The role of competency questions in enterprise engineering. In: Benchmarking—Theory and Practice, pp. 22–31. Springer (1995)
41. Guarino, N., Oberle, D., Staab, S.: Handbook on Ontologies. Springer, Berlin (2009). chap What is an Ontology?, pp. 1–17. International Handbooks on Information Systems
42. Guizzardi, G.: Ontological patterns, anti-patterns and pattern languages for next-generation conceptual modeling. In: Yu, E., Dobbie, G., Jarke, M., Purao, S. (eds.) Conceptual Modeling, pp. 13–27. Springer International Publishing, Cham (2014)
43. Hampe, B.: Image schemas in cognitive linguistics: Introduction. In: Hampe, B., Grady, J.E. (eds.) From Perception to Meaning: Image Schemas in Cognitive Linguistics, pp. 1–14. Walter de Gruyter (2005)
44. Hedblom, M.M.: Image Schemas and Concept Invention: Cognitive, Logical and Linguistic Investigations. PhD thesis, Otto-von-Guericke University Magdeburg (2018)
45. Hedblom, M.M., Kutz, O., Neuhaus, F.: Choosing the right path: Image schema theory as a foundation for concept invention. J. Artif. Gen. Intell. **6**(1), 21–54 (2015)
46. Hedblom, M.M., Kutz, O., Neuhaus, F.: Image schemas as families of theories. In: Besold, T.R., Kühnberger, KU., Schorlemmer, M., Smaill, A. (eds.) Proceedings of the Workshop "Computational Creativity, Concept Invention, and General Intelligence" 2015, Institute of Cognitive Science, Publications of the Institute of Cognitive Science, vol. 2, pp. 19–33 (2015)
47. Hedblom, M.M., Kutz, O., Neuhaus, F.: Image schemas in computational conceptual blending. Cogn. Syst. Res. **39**, 42–57 (2016)
48. Hedblom, M.M., Kutz, O., Mossakowski, T., Neuhaus, F.: Between Contact and Support: Introducing a Logic for Image Schemas and Directed Movement, pp. 256–268. Springer International Publishing, Cham (2017)
49. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proceedings of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR2006), pp. 57–67. AAAI Press (2006)
50. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in OWL ontologies. Web Semantics: Science, Services and Agents on the World Wide Web **3**(4), 268–293 (2005)
51. Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.C.: Repairing unsatisfiable concepts in OWL ontologies. In: ESWC, vol. 6, pp. 170–184. Springer (2006)
52. Koestler, A.: The Act of Creation. Macmillan (1964)
53. Kuhn, W.: An image-schematic account of spatial categories. In: Winter, S., Duckham, M., Kulik, L., Kuipers, B. (eds.) Spatial Information Theory, Lecture Notes in Computer Science, vol. 4736, pp. 152–168. Springer (2007)
54. Kutz, O., Normann, I.: Context discovery via theory interpretation. In: Proc. of the IJCAI Workshop on Automated Reasoning about Context and Ontology Evolution ARCOE-09. Pasadena (2009)
55. Kutz, O., Mossakowski, T., Lücke, D.: Carnap, Goguen, and the Hyperontologies: Logical pluralism and heterogeneous structuring in ontology design, vol. 4. special Issue on 'Is Logic Universal? (2010)
56. Kutz, O., Mossakowski, T., Hois, J., Bhatt, M., Bateman, J.: Ontological blending in DOL. In: Besold, T., Kühnberger, K.U., Schorlemmer, M., Smaill, A. (eds.) Computational Creativity, Concept Invention,

and General Intelligence, Proceedings of the 1st International Workshop C3GI@ECAI, Publications of the Institute of Cognitive Science, Osnabrück, vol. 01-2012, Montpellier (2012)
57. Kutz, O., Bateman, J., Neuhaus, F., Mossakowski, T., Bhatt, M.: E pluribus unum: Formalisation, Use-Cases, and Computational Support for Conceptual Blending. In: Computational Creativity Research: Towards Creative Machines. Thinking Machines. Springer, Atlantis (2014)
58. Kutz, O., Neuhaus, F., Mossakowski, T., Codescu, M.: Blending in the hub—towards a collaborative concept invention platform. In: Proc of the 5th International Conference on Computational Creativity. Ljubljana (2014)
59. Kutz, O., Neuhaus, F., Hedblom, M.M., Mossakowski, T., Codescu, M.: Ontology patterns with DOWL: The case of blending. In: Description Logics. CEUR-WS.org. CEUR Workshop Proceedings, vol. 1577 (2016)
60. Kutz, O., Troquard, N., Hedblom, M.M., Porello, D.: The Mouse and the Ball: Towards a cognitively-based and ontologically-grounded logic of agency. In: Proceedings of the 10th International Conference on Formal Ontology in Information Systems (FOIS 2018), pp. 141–148. IOS Press (2018). https://doi.org/10.3233/978-1-61499-910-2-141
61. van der Laag, P.R., Nienhuys-Cheng, S.H.: Completeness and properness of refinement operators in inductive logic programming. J. Logic Program. **34**(3), 201–225 (1998)
62. Lakatos, I.: Proofs and Refutations. Cambridge University Press (1976)
63. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Mach. Learn. **78**(1-2), 203–250 (2010)
64. Lewis, C.I., Langford, C.H.: Symbolic Logic. Appleton-Century-Crofts, New York (1932)
65. Mandler, J.M., Pagán Cánovas, C.: On defining image schemas. Lang. Cogn. **6**(4), 510–532 (2014)
66. Mossakowski, T., Kutz, O.: The onto-logical translation graph. In: WoMO, Frontiers in Artificial Intelligence and Applications, vol. 230, pp. 94–109. IOS Press (2011)
67. Mossakowski, T., Maeder, C., Lüttich, K.: 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2007, Braga, Portugal, March 24 - April 1, 2007, pp. 519–522. Springer, Berlin (2007). chap The Heterogeneous Tool Set, Hets
68. Mossakowski, T., Codescu, M., Neuhaus, F., Kutz, O.: The Road to Universal Logic: Festschrift for the 50th Birthday of Jean-Yves Béziau, Volume II. Springer International Publishing, Cham, chap The Distributed Ontology. Modeling and Specification Language – DOL, pp. 489–520 (2015)
69. Neuhaus, F.: What is an ontology? arXiv:1810.09171 (2018)
70. Neuhaus, F., Kutz, O., Codescu, M., Mossakowski, T.: Fabricating monsters is hard: Towards the automation of conceptual blending. In: Besold, T., Kuehnberger, K.U., Schorlemmer, M., Smaill, A. (eds.) Computational Creativity, Concept Invention, and General Intelligence, Proc.of the 3rd Int.Workshop C3GI@ECAI-14, Publications of the Institute of Cognitive Science, vol. 1–2014, Osnabrück (2014)
71. Normann, I.: Automated Theory Interpretation. PhD thesis Department of Computer Science. Jacobs University, Bremen (2008)
72. Normann, I., Kutz, O.: Ontology Correspondence via Theory Interpretation. In: Workshop on Matching and Meaning AISB-09. Edinburgh (2009)
73. Nuñez, R., Lakoff, G.: Where Mathematics Comes From. Basic Books (2001)
74. Object Management Group: The distributed ontology, modeling, and specification language (DOL). Document Number ptc/2016-02-37. Available at http://www.omg.org/spec/DOL (2016)
75. Ontañón, S., Plaza, E.: Amalgams: A formal approach for combining multiple case solutions. In: Bichindaritz, I., Montani, S. (eds.) Case-Based Reasoning. Research and Development, pp. 257–271. Springer, ICCBR (2010)
76. Ontañón, S., Plaza, E.: Similarity measures over refinement graphs. Mach. Learn. J. **87**(1), 57–92 (2012)
77. Pagán, C.C.: Erotic emissions in Greek poetry: A generic integration network. Cogn. Semiot. **6**, 7–32 (2010)
78. Pease, A., Lawrence, J., Budzynska, K., Corneli, J., Reed, C.: Lakatos-style collaborative mathematics through dialectical, structured and abstract argumentation. Artif. Intell. **246**, 181–219 (2017)
79. Pereira, F.C.: Creativity and Artificial Intelligence: A Conceptual Blending Approach. Mouton de Gruyter (2007)
80. Porello, D., Troquard, N., Confalonieri, R., Galliani, P., Kutz, O., Peñaloza, R.: Repairing socially aggregated ontologies using axiom weakening. In: PRIMA, vol. 10621, pp. 441–449. Springer, Lecture Notes in Computer Science (2017)
81. Porello, D., Troquard, N., Peñaloza, R., Confalonieri, R., Galliani, P., Kutz, O.: Two approaches to ontology integration based on axiom weakening. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-ECAI 2018), pp. 1942–1948 (2018)

82. Porello, D., Kutz, O., Righetti, G., Troquard, N., Galliani, P., Masolo, C.: A toothful of concepts: Towards a theory of weighted concept combination. In: Šimkus, M., Weddell G (eds.) Proceedings of the 32th International Workshop on Description Logics (DL 2019), CEUR-WS, Oslo, Norway, June 18–21, 2019, vol. 2373 (2019)
83. Ren, Y., Parvizi, A., Mellish, C., Pan, J.Z., van Deemter, K., Stevens, R.: Towards competency question-driven ontology authoring. In: The Semantic Web: Trends and Challenges, pp. 752–767. Springer (2014)
84. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. of IJCAI-03, pp. 355–362. Morgan Kaufmann (2003)
85. Schorlemmer, M., Smaill, A., Kühnberger, K.U., Kutz, O., Colton, S., Cambouropoulos, E., Pease, A.: COINVENT: Towards a computational concept invention theory. In: Proceedings of the 5th International Conference on Computational Creativity. Ljubljana (2014)
86. Schorlemmer, M., Confalonieri, R., Plaza, E.: Coherent conceptual blending. In: Computational Creativity, Concept Invention, and General Intelligence, C3GI at ESSLLI 2016 (2016)
87. Schorlemmer, M., Confalonieri, R., Plaza, E.: The Yoneda Path to the Buddhist Monk Blend. In: First International Workshop on Cognition and Ontologies (CAOS 2016), Annecy, France, 6th July (2016)
88. Schulz, S.: E – A Brainiac theorem prover. J. AI Commun. **15**(2/3), 111–126 (2002)
89. Schwering, A., Krumnack, U., Kühnberger, K.U., Gust, H.: Syntactic principles of heuristic-driven theory projection. Cogn. Syst. Res. **10**(3), 251–269 (2009)
90. Skjæveland, M.G., Forssell, H., Klüwer, J.W., Lupp, D.P., Thorstensen, E., Waaler, A.: Pattern-based ontology design and instantiation with reasonable ontology templates. In: 8th Workshop on Ontology Design and Patterns (WOP 2017). CEUR-WS.org, CEUR Workshop Proceedings, vol. 2043 (2017)
91. St Amant, R., Morrison, C.T., Chang, Y.H., Cohen, P.R., Beal, C.: An image schema language. In: International Conference on Cognitive Modeling (ICCM), pp. 292–297 (2006)
92. Thagard, P.: Coherence in Thought and Action. The MIT Press (2000)
93. Troquard, N., Confalonieri, R., Galliani, P., Peñaloza, R., Porello, D., Kutz, O.: Repairing Ontologies via Axiom Weakening. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, pp. 1981–1988 (2018)
94. Turner, M.: The Origin of Ideas, Blending, Creativity, and the Human Spark. Oxford University Press (2014)
95. Veale, T.: From conceptual mash-ups to "bad-ass" blends: A robust computational model of conceptual blending. In: Proc. of the 2012. International Conference on Computational Creativity. Dublin (2012)
96. Veale, T.: From conceptual "mash-ups" to "bad-ass" blends: A robust computational model of conceptual blending. In: Proceedings of the Third International Conference on Computational Creativity, International Association for Computational Creativity, ICCC 2012, pp. 1–8 (2012)
97. Veale, T., Donoghue, D.O.: Computation and blending. Cogn. Ling. **11**(3-4), 253–282 (2000)
98. Walton, L., Worboys, M.: An algebraic approach to image schemas for geographic space. In: Proceedings of the 9th International Conference on Spatial Information Theory COSIT, pp. 357–370. France (2009)
99. Wilson, D., Sperber, D.: Relevance theory. In: Horn, L., Ward, G. (eds.) The Handbook of Pragmatics, pp. 607–632. Blackwell (2002)
100. Ẑnidarŝiĉ, M., Cardoso, A., Gervas, P., Martins, P., Hervas, R., Alves, A., Oliveira, H., Xiao, P., Linkola, S., Toivonen, H., Kranjc, J., Lavraĉ, N.: Computational creativity infrastructure for online software composition: A conceptual blending use case. In: International Conference on Computational Creativity (ICCC16). Paris, France, 1st July (2016)